Artificial Intelligence 342 (2025) 104310

Contents lists available at ScienceDirect

Artificial Intelligence

journal homepage: www.elsevier.com/locate/artint

Lifted inference beyond first-order logic

Sagar Malhotra [a,*], Davide Bizzaro [b,c], Luciano Serafini [b]

[a] TU Wien, Vienna, Austria
[b] Fondazione Bruno Kessler, Trento, Italy
[c] University of Padova, Padova, Italy

ARTICLE INFO

Keywords:
Lifted inference
Enumerative combinatorics
Weighted model counting
First order logic
Directed acyclic graphs
Connected graphs
Counting quantifiers
Statistical relational learning

ABSTRACT

Weighted First Order Model Counting (WFOMC) is fundamental to probabilistic inference in statistical relational learning models. As WFOMC is known to be intractable in general (#P-complete), logical fragments that admit polynomial time WFOMC are of significant interest. Such fragments are called *domain liftable*. Recent works have shown that the two-variable fragment of first order logic extended with counting quantifiers (C^2) is domain-liftable. However, many properties of real-world data, like *acyclicity* in citation networks and *connectivity* in social networks, cannot be modeled in C^2, or first order logic in general. In this work, we expand the domain liftability of C^2 with multiple such properties. We show that any C^2 sentence remains domain liftable when one of its relations is restricted to represent a directed acyclic graph, a connected graph, a tree (resp. a directed tree) or a forest (resp. a directed forest). All our results rely on a novel and general methodology of *counting by splitting*. Besides their application to probabilistic inference, our results provide a general framework for counting combinatorial structures. We expand a vast array of previous results in discrete mathematics literature on directed acyclic graphs, phylogenetic networks, etc.

1. Introduction

Statistical Relational Learning (SRL) [1,2] is concerned with modeling, learning and inferring over relational data. Probabilistic inference and learning in SRL models like Markov Logic Networks (MLN) [3] and Probabilistic Logic Programs (PLP) [4] can be reduced to instances of *Weighted First Order Model Counting* (WFOMC) [5,6]. WFOMC is the task of computing the weighted sum of the models of a First Order Logic (FOL) sentence Φ over a finite domain of size n. Formally,

$$\text{WFOMC}(\Phi, \text{W}, n) = \sum_{\omega \models \Phi} \text{W}(\omega) \tag{1}$$

where W is a *weight function* that associates a real number to each interpretation ω. Fragments of FOL that admit polynomial time WFOMC w.r.t the domain cardinality n are known as *domain-liftable* [7] or equivalently are said to admit *lifted-inference*. Recent works have also explored WFOMC as a tool for dealing with combinatorics, providing closed form formulae for counting combinatorial structures [8–11] and identifying novel combinatorial sequences [12,13]. This generality of WFOMC applications has led to significant interest in FOL fragments that are domain liftable [14–16].

* Corresponding author.
E-mail address: sagar.malhotra@tuwien.ac.at (S. Malhotra).

https://doi.org/10.1016/j.artint.2025.104310
Received 8 September 2023; Received in revised form 20 February 2025; Accepted 20 February 2025

Available online 24 February 2025

A series of results [8,5] have led to a rather clear picture of domain-liftability in the two-variable fragment of FOL. Especially, these results have shown that any FOL formula in the two-variable fragment is domain-liftable [8]. These positive results are also accompanied by intractability results [8,17], showing that there is an FOL formula in the three-variable fragment whose (W)FOMC can not be computed in polynomial time. Hence, a significant effort has been made towards expanding the domain-liftability of the two-variable fragment of FOL with additional constraints [9,18,11,19,20]. However, many features of real-world data are still not captured in these domain-liftable fragments. Among such constraints are *acyclicity* and *connectivity*. Both of them are ubiquitous features of real-world data [21,22], and have been investigated extensively in combinatorics literature [23–25]. Therefore, extending domain-liftable fragments of FOL with these constraints is of both theoretical and practical interest. However, both acyclicity and connectivity cannot be expressed in FOL [26]. Formally, this means that there is no FOL sentence with fixed number of variables or quantifiers, that can express that a graph is acyclic or connected for arbitrary number of nodes.

In this work, we introduce a general principle of *counting by splitting* for relational structures (Section 4). The principle is based on the simple observation that any interpretation preserves the satisfaction of universally quantified FOL sentences when projected onto a subset of the domain (Proposition 1). Conversely, interpretations of a given universally quantified FO^2 sentence Φ over disjoint sets of domain constants can be expanded (in different ways) to a new interpretation on the union of the sets, such that Φ and some additional desired properties are satisfied (as evidenced in Lemma 1 and 2). Using counting by splitting with existing combinatorial approaches, we expand the domain-liftability of the universally quantified formulas in the two-variable fragment of FOL (FO^2) with directed acyclicity (Section 5) and connectivity constraints (Section 6). Formally, we show that a purely universally quantified FO^2 sentence, with a distinguished predicate restricted to represent a directed acyclic graph or a connected graph, is domain-liftable. We expand these results to full FO^2, extended with *cardinality constraints*, and *counting quantifiers* (C^2), using existing techniques in the literature [5,18]. The combination of directed acyclicity, connectivity, and counting quantifiers, allow easy extension of domain-liftability of C^2, with tree, directed tree and directed forest constraints (Section 7). Finally, forest constraints are obtained by using tree constraints, and counting by splitting.

Besides its applications to SRL, our work provides a formal language in which any expressed constraint can be counted tractably, offering a valuable tool for solving problems in enumerative combinatorics [27]. Most algorithms in this field are devised on a case-by-case basis, almost independently for each problem. In contrast, WFOMC with graph-theoretical constraints offers a general framework: with it, the main task reduces to formulating a logical expression for each problem — something that is significantly easier than devising individual counting algorithms. We illustrate this point with multiple examples of problems investigated in the combinatorics literature that can be easily expressed as the WFOMC of C^2 formulas with the presented constraints. To the best of our knowledge, we show for the first time that the number of *binary phylogenetic networks* on n nodes can be counted in polynomial time w.r.t the number of nodes (see Example 7).

Before presenting our main results (Sections 4-7), we briefly survey the existing related work in Section 2, and provide the necessary background in Section 3. Finally, in Section 8, we empirically investigate the efficiency and scalability of WFOMC with graph-theoretical constraints as a combinatorial framework, and investigate the increase in expressivity of MLNs with the presented constraints.

2. Related work

WFOMC was initially formalized in [6] and [14] for its applications to SRL frameworks like MLNs [3] and PLPs [4]. Initial works in WFOMC generalized *knowledge compilation* techniques to FOL theories, providing an algorithm for Symmetric-WFOMC over universally quantified FOL theories. The formal definition of lifted inference as FOL theories admitting polynomial time weighted model counting w.r.t the domain cardinality was provided in [7]. This created a deep connection between the investigation of efficient probabilistic inference algorithms and the theoretical analysis of WFOMC. A significant advance in WFOMC techniques came in the form of a WFOMC-preserving skolemization procedure [5], which allowed for the complete two-variable fragment to admit polynomial time WFOMC. These works were followed by extensive complexity analysis of WFOMC [8], providing a closed-form formula for WFOMC in the universally quantified fragment of FO^2. However, these results also showed that in the three-variable fragment of FOL there is a formula whose WFOMC cannot be computed in polynomial time, demonstrating that WFOMC is $\#P_1$-complete in general. This led to significant interest in expanding domain liftable fragments in other directions, for instance by adding functionality constraints [9] (i.e., a relation in the language is a function). A major expansion in domain-liftable fragments came in the form of domain-liftability of C^2 [18,11]. Recent results have focused on expanding the expressivity of C^2 with additional constraints like the linear order axiom [19] and tree axiom [20]. The approaches proposed in this paper — which proves domain-liftability of FO^2 extended with acyclicity, connectivity, forests and trees constraints — are most closely related to [8,11,10]. We also exploit the results from [18] (especially for dealing with cardinality constraints) to expand our results to C^2. Our work is also related to recent results that expand WFOMC of C^2 with Tree axiom [20]. However, the techniques introduced in this paper are different and more general than the ones used in [20]. This is because we are able to obtain trees as an instance of connected graphs with $n-1$ edges, which can be simply encoded in cardinality constraints. Finally, our work also draws significantly from combinatorics literature on DAGs [24], connected graphs [28] and forests [25]. And it provides a new resource for tackling other problems of interest in combinatorics, like the enumeration of phylogenetic networks [29–31].

3. Background

This section overviews basic notation, assumptions and aspects of FOL and WFOMC, revisiting and reformulating existing works — as relevant to our results — on WFOMC in FO^2 and C^2. Additional necessary background is provided at the beginning of each section.

3.1. Basic notation

We use $[n]$ to denote the set of integers $\{1, \ldots, n\}$. Whenever the set of integers $[n]$ is obvious from the context and $m \in [n]$ we will use $[\bar{m}]$ to represent the set $\{m+1, \ldots, n\}$. Bold font letters (e.g. $\boldsymbol{k}$) are used to denote integer vectors, and corresponding regular font letters (with an additional index) for the components of the vectors (e.g. k_i). Hence, we use $\boldsymbol{k} = \langle k_1, ..., k_u \rangle$ to denote a vector of u non-negative integers. Given an integer vector $\boldsymbol{k}$, we use $|\boldsymbol{k}|$ to denote $\sum_{i \in [u]} k_i$, i.e. the sum of all the components of $\boldsymbol{k}$. We will also use the multinomial coefficients denoted by

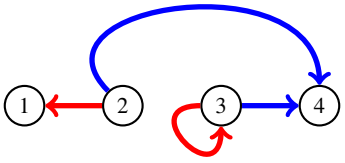
$$\binom{|\boldsymbol{k}|}{k_1, ..., k_u} = \binom{|\boldsymbol{k}|}{\boldsymbol{k}} := \frac{|\boldsymbol{k}|!}{\prod_{i \in [u]} k_i!}$$

When summing two vectors $\boldsymbol{k}' = \langle k'_1, ..., k'_u \rangle$ and $\boldsymbol{k}'' = \langle k''_1, ..., k''_u \rangle$, we always intend element-wise sum, i.e., $\boldsymbol{k}' + \boldsymbol{k}'' = \langle k'_1 + k''_1, ..., k'_u + k''_u \rangle$.

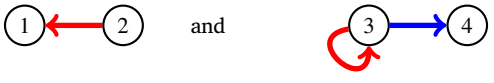
3.2. First-order logic

We assume a function-free FOL language $\mathcal{L}$ defined by a set of variables $\mathcal{V}$ and a set of relational symbols $\mathcal{R}$. We write R/k to denote that the relational symbol R has arity k. For $(x_1, ..., x_k) \in \mathcal{V}^k$ and $R/k \in \mathcal{R}$, we call $R(x_1, ..., x_k)$ an *atom*. A *literal* is an atom or its negation. A *formula* is formed by connecting atoms with boolean operators ($\neg, \vee$ and $\wedge$) and quantifiers of the form $\exists x_i.$ (existential quantification) and $\forall x_i.$ (universal quantification), using FOL syntax rules. The *free* variables of a formula are those that are not bound by a quantifier. We write $\Phi(x_1, \ldots x_k)$ to denote a formula whose free variables are $\{x_1, \ldots, x_k\}$. An FOL formula with no free-variables is called a *sentence*. Hence, a sentence is denoted by a capital Greek letter (e.g. Ψ). The sentences in $\mathcal{L}$ are interpreted over a set of constants called the *domain*. The set of ground atoms (resp. ground literals) are obtained by replacing all the variables in the atoms (resp. literals) of $\mathcal{L}$ with domain constants. Hence, given a predicate R/k, and a domain Δ of size n, we have n^k ground atoms of the form $R(a_1, \ldots, a_k)$, where $(a_1 \ldots a_k) \in \Delta^k$. An *interpretation* ω, on a finite *domain* Δ, is a truth assignment to all the ground atoms. We assume Herbrand semantics [32], hence ω is an interpretation or a model of Ψ if $\omega \models \Psi$ under Herbrand semantics. Given a literal (resp. ground literal) l, we use $pred(l)$ to denote the relational symbol in l. For a subset Δ' of the domain Δ, we use $\omega \downarrow \Delta'$ to denote the partial interpretation induced on Δ'. Hence, $\omega \downarrow \Delta'$ is an interpretation over the ground atoms containing only the domain elements in Δ'. Finally, we use ω_R to represent the partial interpretation of ω restricted to the relation R. When R is binary, ω_R can be seen as a directed graph in which there is an edge from a domain element c to a domain element d whenever $\omega \models R(c, d)$. We can also restrict ω_R to undirected graphs by axiomatizing R to be anti-reflexive and symmetric, by adding the axiom $\forall x. \neg R(x, x) \wedge \forall xy. R(x, y) \rightarrow R(y, x)$.

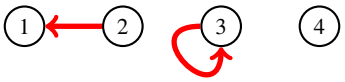
Example 1. Let us have a language with only the two relational symbols R and B, both of arity 2. We represent an interpretation ω as a multi-relational directed graph, where a pair of domain elements (c, d) has a red (resp. blue) directed edge from c to d if and only if $R(c, d)$ (resp. $B(c, d)$) is true in ω. (For interpretation of the references to color please refer to the web version of this article.) Let us take for example the following interpretation ω on $\Delta = [4]$:



Then, $\omega' = \omega \downarrow [2]$ and $\omega'' = \omega \downarrow [\bar{2}]$ are given respectively as



On the other hand, ω_R is obtained by projecting on the predicate R and is given as



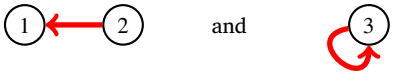
We will also exploit the following proposition about purely universally quantified FOL sentences.

Proposition 1. *Let Ψ be a first order logic sentence such that Ψ is of the form $\forall \boldsymbol{x}.\Phi(\boldsymbol{x})$, where $\boldsymbol{x} = x_1, ..., x_k$ represents the free variables in $\Phi(\boldsymbol{x})$, and $\Phi(\boldsymbol{x})$ is quantifier-free. If ω is an interpretation over a domain Δ, and $\omega \models \Psi$, then $\omega \downarrow \Delta' \models \Psi$ for all $\Delta' \subseteq \Delta$.*

Proof. Under Herbrand semantics, $\omega \models \forall \boldsymbol{x}.\Phi(\boldsymbol{x})$ can be equivalently written as $\omega \models \bigwedge_{\mathbf{c} \in \Delta^k} \Phi(\mathbf{c})$. Hence, $\omega \models \bigwedge_{\mathbf{c} \in \Delta'^k} \Phi(\mathbf{c})$, for any $\Delta' \subseteq \Delta$. Now, $\omega \downarrow \Delta'$ is the truth assignment to all the atoms on the domain Δ'. Hence, $\omega \downarrow \Delta' \models \bigwedge_{\mathbf{c} \in \Delta'^k} \Phi(\mathbf{c})$. Therefore, $\omega \downarrow \Delta' \models \forall \boldsymbol{x}.\Phi(\boldsymbol{x})$. □

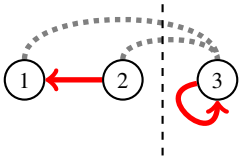
Proposition 1 allows us to split an interpretation over disjoint subsets of the domain while preserving satisfaction of a universally quantified FOL sentence. We will also deal with expanding interpretations from two disjoint subsets of the domain. Given disjoint sets of domain constants Δ' and Δ'', we use $\Delta = \Delta' \uplus \Delta''$ to denote the fact that Δ is the union of the two disjoint sets. If ω' is an interpretation on Δ' and ω'' is an interpretation on Δ'', then we use $\omega' \uplus \omega''$ to denote the partial interpretation on $\Delta' \uplus \Delta''$ obtained by interpreting the ground atoms over Δ' as they are interpreted in ω' and the ground atoms over Δ'' as they are interpreted in ω''. The ground atoms involving domain constants from both Δ' and Δ'' are left un-interpreted in $\omega' \uplus \omega''$. We illustrate this point in the following:

Example 2. Let us have an FOL language with only one relational symbol R, of arity 2. Let $\Delta = [3]$, $\Delta' = [2]$ and $\Delta'' = [\bar{2}] = \{3\}$. We represent an interpretation ω as a directed graph where each pair (c, d) of domain elements has a red edge from c to d if and only if $R(c, d)$ is true in ω. Let us have the following two interpretations ω' and ω'' on the domains $[2]$ and $[\bar{2}]$, respectively:

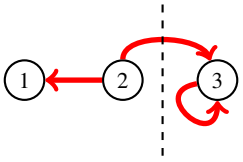


ω' ω''

We now create a partial interpretation $\omega' \uplus \omega''$ as follows:



The gray dotted lines ..... represent the fact that $R(1,3)$, $R(3,1)$, $R(2,3)$ and $R(3,2)$ are not interpreted in $\omega' \uplus \omega''$. A possible extension of $\omega' \uplus \omega''$ is given as



where $R(2,3)$ is interpreted to be true and $R(1,3)$, $R(3,1)$ and $R(3,2)$ are interpreted to be false. We can see that $\omega' \uplus \omega''$ can be extended in 2^4 ways.

3.2.1. FO^2 and its extensions

FO^2 is the fragment of FOL with only two variables. An extension of FO^2 is obtained by introducing *counting quantifiers* of the form $\exists^{=k}$ (there exist exactly k), $\exists^{\geq k}$ (there exist at least k) and $\exists^{\leq k}$ (there exist at most k). This extended fragment is denoted by C^2 [33]. Cardinality constraints are constraints on the cardinality of predicates in an interpretation [18]. For example, $\omega \models \Phi \wedge (|R| \geq 5)$ iff $\omega \models \Phi$ and the sum of the ground atoms in ω, with the predicate R, that are interpreted to be true is at least 5.

3.2.2. Types and tables

We will use the notion of 1-types, 2-type, and 2-tables as presented in [9,34]. A 1-type is a maximally consistent conjunction of literals containing only one variable. For example, in a language with only the relational symbols $U/1$ and $R/2$, both $U(x) \wedge R(x, x)$

and $U(x) \wedge \neg R(x,x)$ are examples of 1-types with variable x. A 2-table is a maximally consistent conjunction of literals containing exactly two distinct variables. Extending the previous example, both $R(x,y) \wedge \neg R(y,x) \wedge (x \neq y)$ and $R(x,y) \wedge R(y,x) \wedge (x \neq y)$ are instances of 2-tables. We assume an arbitrary order on the 1-types and 2-tables. Hence, we use $i(x)$ (resp. $i(y)$) to denote the i^{th} 1-type with variable x (resp. with variable y), and $l(x,y)$ to denote the l^{th} 2-table. A 2-type is a formula of the form $i(x) \wedge j(y) \wedge l(x,y)$, and we use $ijl(x,y)$ to represent it. In a given interpretation ω, we say that a domain constant c realizes the i^{th} 1-type if $\omega \vDash i(c)$, that a pair of domain constants (c,d) realizes the l^{th} 2-table if $\omega \vDash l(c,d)$ and that (c,d) realizes the 2-type $ijl(x,y)$ if $\omega \vDash ijl(c,d)$. We will use u to denote the number of 1-types and b to denote the number of 2-tables in a given FOL language. For instance, in the language with the relational symbols $U/1$ and $R/2$, we have that $u = 2^2$ and $b = 2^2$.

Definition 1 *(1-type Cardinality Vector)*. An interpretation ω is said to have the 1-type cardinality vector $\boldsymbol{k} = \langle k_1, \ldots, k_u \rangle$ if for all $i \in [u]$, it has k_i domain elements c such that $\omega \vDash i(c)$, where $i(x)$ is the i^{th} 1-type. If ω has 1-type cardinality vector $\boldsymbol{k}$, then we say that $\omega \vDash \boldsymbol{k}$.

Notice that in a given interpretation ω each domain element realizes exactly one 1-type. Hence, given a 1-type cardinality vector $\boldsymbol{k}$, $|\boldsymbol{k}|$ is equal to the domain cardinality. Furthermore, for a given $\boldsymbol{k}$ and a fixed pair of 1-types i and j, where $i \leq j$, there are $k_i k_j$ pairs of domain constants (c,d) such that $\omega \vDash i(c) \wedge j(d)$. Similarly, for a given $\boldsymbol{k}$ and a 1-type i, there are $\binom{k_i}{2}$ unordered pairs of distinct domain constants (c,d) such that $\omega \vDash i(c) \wedge i(d)$.

3.3. Weighted first order model counting

In WFOMC as defined in equation (1), we assume that the weight function W does not depend on individual domain constants, which implies that W assigns the same weight to two interpretations which are isomorphic under a permutation of domain elements. Hence, for a domain Δ of size n we can equivalently use $[n]$ as our domain. Furthermore, as common in literature, we will focus on a special class of weight functions, namely *symmetric weight functions*, defined as follows:

Definition 2 *(Symmetric Weight Function)*. Let $\mathcal{G}$ be the set of all ground atoms and $\mathcal{R}$ the set of all relational symbols in a given FOL language. A symmetric weight function associates two real-valued weights, $w : \mathcal{R} \to \mathbb{R}$ and $\bar{w} : \mathcal{R} \to \mathbb{R}$, to each relational symbol in the language. The weight of an interpretation ω is then defined as follows:

$$\mathrm{W}(\omega) := \prod_{\substack{\omega \vDash g \\ g \in \mathcal{G}}} w(pred(g)) \prod_{\substack{\omega \vDash \neg g \\ g \in \mathcal{G}}} \bar{w}(pred(g)). \tag{2}$$

We use $(w, \bar{w})$ to denote a symmetric weight function.

Many techniques [5,18,11] for WFOMC of a sentence Φ require WFOMC-preserving reductions. That is, constructing another sentence Φ' and a new weight function $(w', \bar{w}')$ such that

$$\mathrm{WFOMC}(\Phi, (w, \bar{w}), n) = \mathrm{WFOMC}(\Phi', (w', \bar{w}'), n).$$

A key property desired from a WFOMC-preserving reduction is that adding new formulas to it should not invalidate the reduction. Reductions satisfying this requirement are called *modular*. We formalize this in the following definition:

Definition 3. [5] A reduction $(\Phi, w, \bar{w})$ to $(\Phi', w', \bar{w}')$, where Φ and Φ' are two sentences and $(w, \bar{w})$ and $(w', \bar{w}')$ are two weight functions, is modular if

$$\mathrm{WFOMC}(\Phi \wedge \Lambda, (w, \bar{w}), n) = \mathrm{WFOMC}(\Phi' \wedge \Lambda, (w', \bar{w}'), n) \tag{3}$$

for any sentence Λ.

Intuitively, modular reductions are sound under the presence of another sentence Λ, and any new sentence Λ does not invalidate a modular reduction.

For the rest of the paper, whenever referring to weights, we intend symmetric weights. Hence, we denote $\mathrm{WFOMC}(\Phi, (w, \bar{w}), n)$ without explicitly mentioning the weights, i.e. as $\mathrm{WFOMC}(\Phi, n)$. We will often compute the WFOMC of interpretations with a fixed 1-type cardinality vector, and with a slight abuse of notation denote it as $\mathrm{WFOMC}(\Phi, \boldsymbol{k})$. Formally,

$$\mathrm{WFOMC}(\Phi, \boldsymbol{k}) := \sum_{\omega \vDash \Phi \wedge \boldsymbol{k}} \mathrm{W}(\omega) \tag{4}$$

3.3.1. WFOMC in FO^2

We revisit WFOMC for universally quantified FO^2 formulas, i.e. formulas of the form $\forall xy.\Phi(x,y)$, where $\Phi(x,y)$ is quantifier-free. We define $\Phi(\{x,y\})$ as

$$\Phi(x,x) \wedge \Phi(x,y) \wedge \Phi(y,x) \wedge \Phi(y,y) \wedge (x \neq y)$$

We also define the notion of consistent 2-types as follows:

Definition 4. Given an FO^2 sentence $\forall xy.\Phi(x,y)$, where $\Phi(x,y)$ is quantifier-free, we say that a 2-type $ijl(x,y)$ is consistent with $\forall xy.\Phi(x,y)$ if

$$ijl(x,y) \models \Phi(\{x,y\}) \tag{5}$$

Note that the entailment in equation (5) is checked by assuming a propositional language consisting of only the (constant free) atoms in the FO^2 language.

Example 3. Let $\Phi(x,y) := (A(x) \wedge R(x,y)) \rightarrow A(y)$. The following is an example of a consistent 2-type for the sentence $\forall xy.\Phi(x,y)$:

$$\tau(x,y) := \neg A(x) \wedge R(x,x) \wedge \neg A(y) \wedge R(y,y) \wedge \neg R(x,y) \wedge R(y,x) \wedge (x \neq y)$$

A key idea in analyzing domain-liftability of universally quantified FO^2 formula is that a pair of domain constants (c,d) in an interpretation $\omega \models \forall xy.\Phi(x,y)$ can realize a 2-type $ijl(c,d)$ only if the 2-type is consistent with the formula $\forall xy.\Phi(x,y)$, i.e. only if $ijl(x,y) \models \Phi(\{x,y\})$. We formalize this intuition in the following proposition.

Proposition 2. *Let $\forall xy.\Phi(x,y)$ be an FO^2 sentence where $\Phi(x,y)$ is quantifier-free. Then, $\omega \models \forall xy.\Phi(x,y)$ iff for any pair of distinct domain constants (c,d), such that $\omega \models ijl(c,d)$, we have that $ijl(x,y)$ is consistent with $\forall xy.\Phi(x,y)$, i.e. $ijl(x,y) \models \Phi(\{x,y\})$.*

Proof. If $\omega \models \forall xy.\Phi(x,y)$ and $\omega \models ijl(c,d)$, then $\omega \models \forall xy.\Phi(x,y) \wedge ijl(c,d)$. Now, $ijl(c,d)$ is a complete truth assignment to the ground atoms containing only the domain constants c or d or both. Hence, $\omega \models \forall xy.\Phi(x,y) \wedge ijl(c,d)$ only if $ijl(c,d) \models \Phi(\{c,d\})$, i.e. only if $ijl(x,y) \models \Phi(\{x,y\})$. On the other hand, let ω be such that all pair of domain constants realize only the 2-types $ijl(x,y)$ consistent with $\forall xy.\Phi(x,y)$. It means that $\omega \models \Phi(\{c,d\})$ for all the pairs of domain constants (c,d). Hence, $\omega \models \forall xy.\Phi(x,y)$. □

To facilitate the treatment of WFOMC, we will now introduce some weight parameters associated with an FO^2 language. Consider an FO^2 language $\mathcal{L}$ with symmetric weight function $(w,\bar{w})$, and let $\mathcal{I}$ denote the set of atoms in $\mathcal{L}$ that contain only variables. We define the following weight parameters associated with each 1-type $i(x)$ and 2-table $l(x,y)$:

$$w_i := \prod_{\substack{i(x) \models g \\ g \in \mathcal{I}}} w(pred(g)) \prod_{\substack{i(x) \models \neg g \\ g \in \mathcal{I}}} \bar{w}(pred(g)) \tag{6}$$

and

$$v_l := \prod_{\substack{l(x,y) \models g \\ g \in \mathcal{I}}} w(pred(g)) \prod_{\substack{l(x,y) \models \neg g \\ g \in \mathcal{I}}} \bar{w}(pred(g)) \tag{7}$$

We now present an analytical formula for WFOMC of FO^2 sentences of the form $\forall xy.\Phi(x,y)$ where $\Phi(x,y)$ is quantifier-free. Our presentation is based on the treatment proposed in [8].

Theorem 1 *(reformulated from [8]). Given an FO^2 sentence $\forall xy.\Phi(x,y)$ where $\Phi(x,y)$ is quantifier-free, let us define, for any pair of 1-types (i,j), the quantity $r_{ij} := \sum_{l \in [b]} n_{ijl} v_l$, with n_{ijl} being 1 if $ijl(x,y) \models \Phi(\{x,y\})$ and 0 otherwise. Then, the weighted model count of $\forall xy.\Phi(x,y)$ with 1-types cardinality vector $\boldsymbol{k}$ is given as follows:*

$$\text{WFOMC}(\forall xy.\Phi(x,y), \boldsymbol{k}) = \binom{|\boldsymbol{k}|}{\boldsymbol{k}} \prod_{i \in [u]} w_i^{k_i} \prod_{i \leq j \in [u]} r_{ij}^{\boldsymbol{k}(i,j)} \tag{8}$$

where $\boldsymbol{k}(i,j)$ is defined as

$$\boldsymbol{k}(i,j) := \begin{cases} \frac{k_i(k_i-1)}{2} & \text{if } i = j \\ k_i k_j & \text{otherwise} \end{cases}$$

Proof. In a 1-type cardinality vector $\boldsymbol{k}$, and k_i represents the number of domain constants that realize the 1-type i. Since any domain constant realizes one and only one 1-type in any given interpretation, there are exactly $\binom{|\boldsymbol{k}|}{\boldsymbol{k}}$ ways of assigning 1-types to $|\boldsymbol{k}|$ domain constants. Suppose that a domain constant c realizes the i^{th} 1-type for an interpretation $\omega \models \boldsymbol{k}$; then $i(c)$ contributes to the weight of the interpretation with the weight w_i, multiplicatively. Therefore, the contribution due to 1-type realizations is given by $\prod_{i \in [u]} w_i^{k_i}$ for any interpretation $\omega \models \boldsymbol{k}$. Summing over all such interpretations gives us the factor of $\binom{|\boldsymbol{k}|}{\boldsymbol{k}} \prod_{i \in [u]} w_i^{k_i}$.

Now consider an interpretation ω and a pair of distinct domain constants (c, d) such that $\omega \models i(c) \wedge j(d)$. Using Proposition 2, we know that (c, d) realizes the 2-table $l(c, d)$ if and only if $ijl(x, y) \models \Phi(\{x, y\})$. Therefore, in an arbitrary interpretation ω, the multiplicative weight contribution due to the realization of the l^{th} 2-table by a pair of constants (c, d) such that $\omega \models i(c) \wedge j(d)$ is given by $n_{ijl}v_l$. Also, each ordered pair of constants can realize exactly one and only one 2-table. Hence, the sum of the weights of the possible 2-table realizations of a pair of domain constants (c, d) such that $i(c)$ and $j(d)$ are true is given as $r_{ij} = \sum_l n_{ijl}v_l$. Furthermore, given the 1-type assignments $i(c)$ and $j(d)$, the ordered pair (c, d) can realize 2-tables independently of all other domain constants. Since there are $\boldsymbol{k}(i, j)$ possible such pairs, the contribution given by the realization of 2-tables over all the interpretations $\omega \models \boldsymbol{k}$ is $\prod_{i \leq j \in [u]} r_{ij}^{\boldsymbol{k}(i,j)}$. □

Equation (8) can be computed in polynomial time w.r.t. the domain cardinality n, and there are only polynomially many $\boldsymbol{k}$ w.r.t. n. Hence, $\text{WFOMC}(\forall xy.\Phi(x, y), n)$, which is equal to

$$\sum_{|\boldsymbol{k}|=n} \text{WFOMC}(\forall xy.\Phi(x, y), \boldsymbol{k}) \tag{9}$$

can be computed in polynomial time w.r.t. domain size n. Furthermore, [5] shows that any FOL formula with existential quantification can be modularly reduced to a WFOMC preserving universally quantified FO^2 formula, with additional new predicates and negative weights. Hence, showing that FO^2 is domain-liftable.

Remark 1. Note that the number of distinct $\boldsymbol{k}$ in equation (9) can be super-exponentially many w.r.t the number of symbols in the language. However, as common in domain-liftability literature, in this paper we are only concerned with computational complexity of WFOMC w.r.t. the domain cardinality n.

3.3.2. WFOMC in C^2

[11] and [18] show that WFOMC in C^2 can be reduced to WFOMC in FO^2 with cardinality constraints. The key result that leads to domain liftability of C^2, is the following:

Theorem 2. *([18], slightly reformulated) Let Φ be a first-order logic sentence and let Γ be an arbitrary cardinality constraint. Then,* $\text{WFOMC}(\Phi \wedge \Gamma, \boldsymbol{k})$ *can be computed in polynomial time with respect to the domain cardinality, relative to the* $\text{WFOMC}(\Phi, \boldsymbol{k})$ *oracle.*

In order to prove Theorem 2, the proof in [18] relies on polynomial interpolation. We provide an easier presentation of the proof using Gauss-elimination based polynomial interpolation in the appendix.

Remark 2. In the proof presented in the appendix (and in [18]), the first-order definability of Φ is never invoked. This property has also been exploited for imposing cardinality constraints with tree axiom in [20].

Theorem 2 extends domain-liftability of any sentence Φ to its domain-liftability with cardinality constraints. We now move to domain-liftability in C^2:

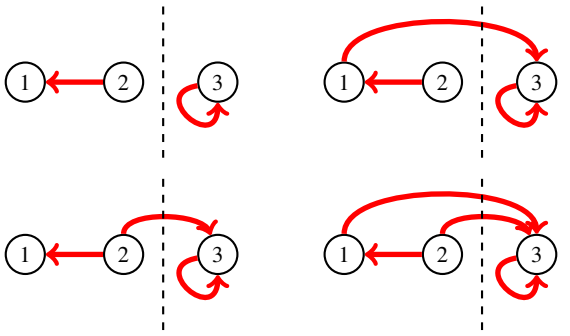
Theorem 3 *([18]). The fragment of first-order logic with two variables and counting quantifiers is domain-liftable.*

The key idea behind Theorem 3 is that WFOMC of a C^2 sentence Φ can be converted to a problem of WFOMC of an FO^2 sentence Φ' with cardinality constraints Γ on an extended vocabulary with additional weights for the new predicates (the new predicates are weighted 1 or -1). We refer the reader to [18] and [11] for the detailed treatment of Theorem 3. However, for our purposes it is important to note that this transformation is modular. The modularity of the WFOMC procedure, as presented in [18], has also been exploited to demonstrate domain-liftability of C^2 extended with Tree axiom [20] and Linear Order axiom [19].

4. Main approach: counting by splitting

As shown in Example 2, the possible ways of *merging* two interpretations ω' and ω'', on disjoint domains Δ' and Δ'', can be efficiently computed. Moreover, additional constraints can be imposed on the way the interpretations are merged. Such constraints, for instance, may express statements like "for the binary predicate R, there is no R-edge going from Δ'' to Δ'". Formally, we express this constraint as $\forall x \in \Delta'\ \forall y \in \Delta''.\neg R(y, x)$.

Example 4. Among the 2^4 possible extensions of $\omega' \uplus \omega''$ in Example 2, the following 2^2 satisfy the additional constraint $\forall x \in [2]\, \forall y \in [\bar{2}].\neg R(y, x)$:



A key insight of the paper is that such merging can also be done for disjoint interpretations of a universally quantified FO^2 formula, while preserving the satisfaction of such formula. This is formalized in the following lemma.

Lemma 1. *Let $\Phi(x,y)$ and $\Theta(x,y)$ be quantifier-free FO^2 formulas. Let ω' and ω'' be interpretations satisfying $\forall xy.\Phi(x,y)$ on two disjoint domains Δ' and Δ'' (respectively). We define $r_{ij} := \sum_{l\in[b]} n_{ijl} v_l$, where n_{ijl} is 1 if $ijl(x,y) \models \Phi(\{x,y\}) \wedge \Theta(x,y)$ and 0 otherwise, and v_l is defined in equation (7). Then, the sum of the weights of the extensions ω of $\omega' \uplus \omega''$ such that*

- $\omega \models \forall xy.\Phi(x,y)$
- $\omega \models \forall x \in \Delta' \; \forall y \in \Delta''.\Theta(x,y)$

is given as

$$\mathrm{w}(\omega')\mathrm{w}(\omega'') \prod_{i,j\in[u]} r_{ij}^{k'_i k''_j} \tag{10}$$

where k'_i and k''_j are the number of domain constants realizing the i^{th} and j^{th} 1-types in ω' and ω'' respectively.

Proof. In order to obtain an interpretation ω from $\omega' \uplus \omega''$, we only need to extend $\omega' \uplus \omega''$ with interpretations of the ground-atoms containing pairs $(c,d) \in \Delta' \times \Delta''$. For a given pair $(c,d) \in \Delta' \times \Delta''$, let i and j be the 1-types such that $\omega' \models i(c)$ and $\omega'' \models j(d)$. Since we want ω to be a model of $\forall xy.\Phi(x,y)$, if $\omega \models l(c,d)$, then by Proposition 2 we must have that $ijl(c,d) \models \Phi(\{c,d\})$. Moreover, since we want ω to satisfy the condition $\forall x \in \Delta' \; \forall y \in \Delta''.\Theta(x,y)$, we must have that $ijl(c,d) \models \Theta(c,d)$. Therefore, (c,d) can realize the l^{th} 2-table if and only if $n_{ijl} = 1$. The multiplicative weight contribution to the weight of a given extension ω of $\omega' \uplus \omega''$ due to (c,d) realizing the l^{th} 2-table is given as $n_{ijl} v_l$. Furthermore, (c,d) realizes the 2-tables mutually-exclusively. Also, the weight contribution of the 2-table realizations of (c,d) only depends on the 1-types of c and d and is independent of all other domain constants. Hence, $r_{ij} = \sum_{l\in[b]} n_{ijl} v_l$ is the multiplicative weight contribution given by the interpretation of all the possible ground atoms containing a pair $(c,d) \in \Delta' \times \Delta''$ with c realizing the i^{th} 1-type in ω' and j realizing the j^{th} 1-type in ω''. Since there are k'_i domain elements c realizing the i^{th} 1-type in ω', and k''_j domain elements d realizing the j^{th} 1-type in ω'', the weight contribution of the interpretation of all the possible ground atoms containing a pair $(c,d) \in \Delta' \times \Delta''$ is given by $\prod_{i,j\in[u]} r_{ij}^{k'_i k''_j}$. Furthermore, since ω' and ω'' interpret two disjoint sets of ground atoms, their weights $\mathrm{w}(\omega')$ and $\mathrm{w}(\omega'')$ contribute independently to each $\mathrm{w}(\omega)$. □

Lemma 1 provides us a method to compute the (weighted sum of the) number of ways to merge two FO^2 interpretations on two disjoint domains. We will now use Lemma 1 to compute weighted model count of FO^2 formulas, where the models are restricted to obey additional constraints (potentially inexpressible in FOL) on disjoint subsets of the domain.

Lemma 2 *(Counting by Splitting). Let $\Phi(x,y)$ and $\Theta(x,y)$ be quantifier-free FO^2 formulas. Let $\Psi' := \forall xy.\Phi(x,y) \wedge axiom'$ and $\Psi'' := \forall xy.\Phi(x,y) \wedge axiom''$, where $axiom'$ and $axiom''$ denote arbitrary constraints – potentially inexpressible in first-order logic. Let $[n]$ be the domain, and let $m \leq n$ be a natural number. Let $\Psi_{[m]}$ denote the conjunction of the following four conditions on interpretations ω:*

C1: $\omega \downarrow [m] \models axiom'$
C2: $\omega \downarrow [\bar{m}] \models axiom''$
C3: $\omega \models \forall x \in [m] \; \forall y \in [\bar{m}].\Theta(x,y)$
C4: $\omega \models \forall xy.\Phi(x,y)$

Then, the weighted model count of the interpretations ω satisfying both $\Psi_{[m]}$ and the cardinality vector $\boldsymbol{k}$ is given as:

$$\mathrm{WFOMC}(\Psi_{[m]}, \boldsymbol{k}) = \sum_{\substack{\boldsymbol{k'}+\boldsymbol{k''}=\boldsymbol{k} \\ |\boldsymbol{k'}|=m}} \mathrm{WFOMC}(\Psi', \boldsymbol{k'})\mathrm{WFOMC}(\Psi'', \boldsymbol{k''}) \prod_{i,j\in[u]} r_{ij}^{k'_i k''_j} \tag{11}$$

Table 1
Axioms used for counting by splitting for different constraints.

Constraint	$axiom'$	$axiom''$	$\Theta(x,y)$
$DAG(R)$	$\forall xy.\neg R(x,y)$	$DAG(R)$	$\neg R(y,x)$
$Connected(R)$	$Connected(R)$	$\top$	$\neg R(x,y)$
$Forest(R)$	$Tree(R)$	$Forest(R)$	$\neg R(x,y)$

where $r_{ij} := \sum_{l\in[b]} n_{ijl} v_l$, *and* n_{ijl} *is* 1 *if* $ijl(x,y) \models \Phi(\{x,y\}) \wedge \Theta(x,y)$ *and* 0 *otherwise, and* v_l *is as defined in equation* (7).

Proof. Let $\omega \models \Psi_{[m]}$. Using condition C4 and Proposition 1, we have that $\omega \downarrow [m] \models \forall xy.\Phi(x,y)$. Furthermore, using condition C1, we have that $\omega \downarrow [m] \models axiom'$. Hence, $\omega \downarrow [m] \models \Psi'$, and similarly $\omega \downarrow [\bar{m}] \models \Psi''$. The WFOMC of Ψ' on $[m]$ with 1-type cardinality vector $\boldsymbol{k}'$ is $\text{WFOMC}(\Psi', \boldsymbol{k}')$. Similarly, the WFOMC of Ψ'' on $[\bar{m}]$ with 1-type cardinality vector $\boldsymbol{k}''$ is $\text{WFOMC}(\Psi'', \boldsymbol{k}'')$. For a given pair of models ω' and ω'' counted respectively in $\text{WFOMC}(\Psi', \boldsymbol{k}')$ and $\text{WFOMC}(\Psi, \boldsymbol{k}'')$, the weighted sum of extensions $\omega \models \Psi_{[m]}$ is given by equation (10), due to Lemma 1. Summing over extensions of all such possible pairs of ω' and ω'', we get:

$$\text{WFOMC}(\Psi', \boldsymbol{k}')\text{WFOMC}(\Psi'', \boldsymbol{k}'') \prod_{i,j\in[u]} r_{ij}^{k_i' k_j''} \tag{12}$$

Expression (12) gives us the WFOMC of the models ω such that $\omega \models \Psi_{[m]}$, $\omega \downarrow [m] \models \boldsymbol{k}'$ and $\omega \downarrow [\bar{m}] \models \boldsymbol{k}''$. To compute the WFOMC of the models ω such that $\omega \models \Psi_{[m]} \wedge \boldsymbol{k}$, we have to sum expression (12) over all possible decompositions of the 1-type cardinality vector $\boldsymbol{k}$ into $\boldsymbol{k}'$ and $\boldsymbol{k}''$, where $|\boldsymbol{k}'| = m$. □

The techniques developed in this paper will use counting by splitting with different instantiations of $axiom'$, $axiom''$ and $\Theta(x,y)$. Table 1 summarizes how we will instantiate them when providing the algorithms for WFOMC with DAG, connected graph and forest constraints (as given respectively in Definitions 5, 8 and 13, and making use of Remarks 3, 4 and 8).

5. WFOMC with DAG axiom

A *Directed Acyclic Graph (DAG)* is a directed graph such that starting from an arbitrary node i and traversing an arbitrary path along directed edges we would never arrive back at node i. We present a recursive formula based on the principle of inclusion-exclusion for WFOMC of an FO^2 sentence, say Φ, where a special predicate, say R, is axiomatized to be a DAG, i.e. we count the models $\omega \models \Phi$ such that ω_R represents a DAG. We begin by revisiting the principle of inclusion-exclusion and the recursive formula for counting DAGs on n nodes, as presented in [24]. We identify key observations that lead to the formula for counting DAGs and exploit analogous ideas for WFOMC with the DAG axiom. We then expand the DAG axiom with *source* and *sink* constraints. Our results also shed a new light on counting phylogenetic networks [29–31] – a widely investigated problem in combinatorics and mathematical biology.

5.1. Principle of inclusion-exclusion

Given a set of finite sets $\{A_i\}_{i\in[n]}$, let $A_J := \bigcap_{j\in J} A_j$ for any subset $J \subseteq [n]$. The *principle of inclusion-exclusion (PIE)* states that:

$$\left|\bigcup_i A_i\right| = \sum_{\emptyset\neq J\subseteq[n]} (-1)^{|J|+1} \left|A_J\right| \tag{13}$$

If the cardinality of the intersections A_J only depends on the cardinality of J, then the formula can be simplified. In this case, all A_J's with $|J| = m$ have cardinality equal to $|A_{[m]}|$. Since for any $m \in [n]$ there are $\binom{n}{m}$ sets A_J with $|J| = m$, equation (13) reduces to:

$$\left|\bigcup_i A_i\right| = \sum_{m=1}^{n} (-1)^{m+1} \binom{n}{m} \left|A_{[m]}\right| \tag{14}$$

The principle of inclusion-exclusion can be easily extended to the case when the sets A_i contain weighted FOL interpretations and we want the weighted sum of all the interpretations in $\bigcup_i A_i$. Indeed, if $\text{W}(A_i)$ denotes the weighted sum of all the interpretations in A_i, then the PIE reduces to:

$$\text{W}\left(\bigcup_i A_i\right) = \sum_{\emptyset\neq J\subseteq[n]} (-1)^{|J|+1} \text{W}(A_J) \tag{15}$$

Analogously, when $\text{W}(A_J) = \text{W}(A_{J'})$ for all J, J' with the same cardinality, then:

$$\text{W}\left(\bigcup_i A_i\right) = \sum_{m=1}^{n} (-1)^{m+1} \binom{n}{m} \text{W}(A_{[m]}) \tag{16}$$

Algorithm 1 Number of DAG on n nodes.

1: **Input**: n
2: **Output**: a_n
3: $A[0] \leftarrow 1$
4: **for** $i = 1$ to n **do**
5: $\quad A[i] \leftarrow \sum_{l=0}^{i-1} (-1)^{i-l+1} \binom{i}{l} 2^{l(i-l)} A[l]$
6: **end for**
7: **return** $A[n]$

5.2. Counting directed acyclic graphs

We now derive the formula for counting DAGs as presented in [24]. Without loss of generality, we assume the set of nodes to be $[n]$. For each $i \in [n]$, let A_i be the set of DAGs on $[n]$ for which the node i has indegree zero. Since every DAG has at least one node with indegree zero, we have that the total number of DAGs a_n is equal to $|\bigcup_{i\in[n]} A_i|$. The set of DAGs such that all nodes in $J \subseteq [n]$ have indegree zero is given by $A_J := \bigcap_{j\in J} A_j$. The cardinality of A_J depends only on the cardinality of J, and not on the individual elements in J. Hence, we can assume w.l.o.g that $J = [m]$, where $m = |J|$. We now derive a method for computing $\left|A_{[m]}\right|$. We make the following key observations:

- Observation 1. If $\omega \in A_{[m]}$, then there are no edges between the nodes in $[m]$, as otherwise a node in $[m]$ will have a non-zero indegree. In other words, only directed edges from $[m]$ to $[\bar{m}]$ or within $[\bar{m}]$ are allowed.
- Observation 2. If $\omega \in A_{[m]}$, then the subgraph of ω restricted to $[\bar{m}]$, i.e. $\omega \downarrow [\bar{m}]$, is a DAG. And the subgraph of ω restricted to $[m]$ is just an empty graph, i.e. the set of isolated nodes $[m]$ with no edges between them.
- Observation 3. Any DAG on $[\bar{m}]$ can be extended to $2^{m(n-m)}$ DAGs in $A_{[m]}$. This is because DAGs in $A_{[m]}$ have no edges between the nodes in $[m]$; they only have outgoing edges from $[m]$ to $[\bar{m}]$. For extending a given DAG on $[\bar{m}]$ to a DAG in $A_{[m]}$, we have two choices for each pair of nodes in $[m] \times [\bar{m}]$: we can either draw an out-going edge from $[m]$ to $[\bar{m}]$ or not. Hence, there are $2^{|[m]\times[\bar{m}]|} = 2^{m(n-m)}$ ways to extend a given DAG on $[\bar{m}]$ to a DAG in $A_{[m]}$.

The number of possible DAGs on $[\bar{m}]$ is a_{n-m}. Due to Observation 3, we have that $A_{[m]}$ has $2^{m(n-m)} a_{n-m}$ DAGs obtained by extending the DAGs on $[\bar{m}]$. Furthermore, due to Observation 1 and Observation 2, these are all the possible DAGs in $A_{[m]}$. Hence, $|A_{[m]}| = 2^{m(n-m)} a_{n-m}$. Now we can repeat this argument for any m-sized subset of $[n]$: if $|J| = |J'| = m$, then $|A_J| = |A_{J'}| = 2^{m(n-m)} a_{n-m}$. Using the principle of inclusion-exclusion as given in equation (14), we have that:

$$a_n = \sum_{m=1}^{n} (-1)^{m+1} \binom{n}{m} 2^{m(n-m)} a_{n-m} \tag{17}$$

Notice that, after replacing $n - m$ with l, the equation becomes:

$$a_n = \sum_{l=0}^{n-1} (-1)^{n-l+1} \binom{n}{l} 2^{l(n-l)} a_l \tag{18}$$

The change of variable allows us to write a bottom-up algorithm for counting DAGs, given in Algorithm 1. Based on this algorithm, it can be seen that a_n can be computed in polynomial time w.r.t n. Indeed, the **for** loop in line 4 runs n times, and in line 5 we compute a_i (and store it as $A[i]$) using equation (18), which requires only polynomially many operations in n.

5.3. WFOMC with DAG axiom

Definition 5. Let R be a binary predicate. We say that an interpretation ω is a model of $Acyclic(R)$, and write $\omega \vDash Acyclic(R)$, if ω_R forms a DAG.

Remark 3. Our goal is to compute the WFOMC of $\Psi := \forall xy.\Phi(x,y) \wedge Acyclic(R)$, where $\Phi(x,y)$ is a quantifier-free FO^2 formula. By definition, this entails that ω_R forms an antireflexive relation of R. Hence, we can assume without loss of generality that $\forall xy.\Phi(x,y) \vDash \forall x.\neg R(x,x)$.

W.l.o.g., we assume that the domain is $[n]$. We will now redefine $\Psi_{[m]}$ for the purposes of WFOMC with DAG axiom.

Definition 6. Let $\Psi := \forall xy.\Phi(x,y) \wedge Acyclic(R)$, where $\Phi(x,y)$ is a quantifier-free FO^2 formula such that $\forall xy.\Phi(x,y) \vDash \forall x.\neg R(x,x)$. For any $m \leq n$, we say that an interpretation ω is a model of $\Psi_{[m]}$, and write $\omega \vDash \Psi_{[m]}$, if ω is a model of Ψ on $[n]$ and the domain elements $[m]$ have indegree zero in ω_R.

It can be shown that $\Psi_{[m]}$ in Definition 6 corresponds to an instantiation of $\Psi_{[m]}$ in Lemma 2, where we instantiate $axiom'$, $axiom''$ and $\Theta(x,y)$ as follows:

- $axiom' := \forall xy.\neg R(x,y)$
- $axiom'' := Acyclic(R)$
- $\Theta(x,y) := \neg R(y,x)$

A formal proof of this correspondence is in the appendix (Lemma 3). With this specific instantiation of $\Psi_{[m]}$, we can use Lemma 2 and get the following proposition.

Proposition 3. *Let $\Psi := \forall xy.\Phi(x,y) \wedge Acyclic(R)$ and $\Psi' := \forall xy.\Phi(x,y) \wedge \neg R(x,y)$, where $\Phi(x,y)$ is a quantifier-free FO^2 formula, such that $\forall xy.\Phi(x,y) \models \forall x.\neg R(x,x)$. Then:*

$$\text{WFOMC}(\Psi_{[m]}, \boldsymbol{k}) = \sum_{\substack{\boldsymbol{k'}+\boldsymbol{k''}=\boldsymbol{k} \\ |\boldsymbol{k'}|=m}} \text{WFOMC}(\Psi', \boldsymbol{k'})\text{WFOMC}(\Psi, \boldsymbol{k''}) \prod_{i,j\in[u]} r_{ij}^{k'_i k''_j} \tag{19}$$

where $r_{ij} := \sum_l n_{ijl} v_l$, and n_{ijl} is 1 if $ijl(x,y) \models \Phi(\{x,y\}) \wedge \neg R(y,x)$ and 0 otherwise.

Similarly to equation (17), we can use $\text{WFOMC}(\Psi_{[m]}, \boldsymbol{k})$ and the principle of inclusion-exclusion for computing the WFOMC for $\Psi := \forall xy.\Phi(x,y) \wedge Acyclic(R)$.

Proposition 4. *Let $\Psi := \forall xy.\Phi(x,y) \wedge Acyclic(R)$, where $\Phi(x,y)$ is a quantifier-free FO^2 formula, such that $\forall xy.\Phi(x,y) \models \forall x.\neg R(x,x)$. Then:*

$$\text{WFOMC}(\Psi, \boldsymbol{k}) = \sum_{m=1}^{|\boldsymbol{k}|} (-1)^{m+1} \binom{|\boldsymbol{k}|}{m} \text{WFOMC}(\Psi_{[m]}, \boldsymbol{k}) \tag{20}$$

Proof. The proof idea is very similar to the case for counting DAGs as given in equation (17). W.l.o.g., we are assuming the domain to be $[n]$, so $|\boldsymbol{k}| = n$. Let A_i be the set of models ω of Ψ, such that ω has 1-type cardinality $\boldsymbol{k}$ and the domain element i has zero R-indegree in ω_R. Since every DAG has at least one node with zero R-indegree, our goal is to compute $\text{W}(\cup_{i\in[n]} A_i)$. For any $J \subseteq [n]$, let $A_J := \bigcap_{j\in J} A_j$. For any J and J' such that $|J| = |J'|$, we always have that $|A_J| = |A_{J'}|$. Hence, using the principle of inclusion-exclusion as given in equation (16), we have that:

$$\text{WFOMC}(\Psi, \boldsymbol{k}) = \sum_{m=1}^{|\boldsymbol{k}|} (-1)^{m+1} \binom{n}{m} \text{W}(A_{[m]}) \tag{21}$$

Now, $A_{[m]}$ is the set of models such that domain elements in $[m]$ have zero R-indegree. Hence, $\text{W}(A_{[m]}) = \text{WFOMC}(\Psi_{[m]}, \boldsymbol{k})$. □

We make a change of variable in equation (20) — similar to the one from (17) to (18) — by replacing m with $|\boldsymbol{k}| - l$. We obtain the following equation:

$$\text{WFOMC}(\Psi, \boldsymbol{k}) = \sum_{l=0}^{|\boldsymbol{k}|-1} (-1)^{|\boldsymbol{k}|-l+1} \binom{|\boldsymbol{k}|}{l} \text{WFOMC}(\Psi_{[|\boldsymbol{k}|-l]}, \boldsymbol{k}) \tag{22}$$

We provide pseudocode for evaluating equation (22) in Algorithm 2, namely WFOMC-DAG. We now analyze how WFOMC-DAG works, and show that it runs in polynomial time with respect to domain cardinality $|\boldsymbol{k}| = n$.

WFOMC-DAG takes as input $\Psi = \forall xy.\Phi(x,y) \wedge Acyclic(R)$ and $\boldsymbol{k}$ and it returns $\text{WFOMC}(\Psi, \boldsymbol{k})$. In line 3, an array A with u indices is initiated, and $A[\mathbf{0}]$ is assigned the value 1, where $\mathbf{0}$ corresponds to the u-dimensional zero vector. The for loop in lines $5-7$ incrementally computes $\text{WFOMC}(\Psi, \boldsymbol{p})$, where the loop runs over all u-dimensional integer vectors $\boldsymbol{p}$, such that $p_i \leq k_i$, in lexicographical order. The number of possible $\boldsymbol{p}$ vectors is bounded by n^u. Hence, the for loop in line 5 runs at most n^u iterations. In line 6, we compute $\text{WFOMC}(\Psi, \boldsymbol{p})$ as given in equation (22). Also in line 6, the function $\overline{\text{WFOMC}}(\Psi_{[m]}, \boldsymbol{p})$ — that computes $\text{WFOMC}(\Psi_{[m]}, \boldsymbol{p})$ — is called at most $|\boldsymbol{p}| - 1$ times, which is bounded above by n. $A[\boldsymbol{p}]$ stores the value $\text{WFOMC}(\Psi, \boldsymbol{p})$. In the function $\overline{\text{WFOMC}}(\Psi_{[m]}, \boldsymbol{s})$, the number of iterations in the for loop (line 12) is bounded above by n^{2u}. And $\text{WFOMC}(\Psi', \boldsymbol{s'})$ is an FO^2 WFOMC problem, again computable in polynomial time. Hence, the algorithm WFOMC-DAG runs in polynomial time w.r.t domain cardinality. Notice that since loop 5-7 runs in lexicographical order, the $A[\boldsymbol{s''}]$ required in the function $\overline{\text{WFOMC}}(\Psi_{[m]}, \boldsymbol{s})$ are always already stored in A.

There are only polynomially many $\boldsymbol{k}$ w.r.t domain cardinality. Hence, computing $\text{WFOMC}(\Psi, \boldsymbol{k})$ over all possible $\boldsymbol{k}$ values, we can compute $\text{WFOMC}(\Psi, n)$ in polynomial time w.r.t domain cardinality. Using the modular WFOMC-preserving Skolemization process as provided in [5], we can extend this result to prove the domain liftability of the entire FO^2 fragment, with DAG axiom.

Using Theorem 2 and Remark 2, we can also extend domain-liftability of FO^2, with DAG axiom and cardinality constraints. Finally, since WFOMC of any C^2 formula can be modularly reduced to WFOMC of an FO^2 formula with cardinality constraints [18], we have the following theorem:

Theorem 4. *Let $\Psi := \Phi \wedge Acyclic(R)$, where Φ is a C^2 sentence. $\text{WFOMC}(\Psi, n)$ can be computed in polynomial time with respect to the domain cardinality.*

Algorithm 2 WFOMC-DAG.

Input: $\Psi, \boldsymbol{k}$
Output: $\text{WFOMC}(\Psi, \boldsymbol{k})$
$A[\mathbf{0}] \leftarrow 1$ ▷ A has u indices
▷ $\mathbf{0} = \langle 0, ..., 0 \rangle$
for $\mathbf{0} < \boldsymbol{p} \leq \boldsymbol{k}$ where $\boldsymbol{p} \in \mathbb{N}_0^u$ **do** ▷ Lexical order
  $A[\boldsymbol{p}] \leftarrow \sum_{l=0}^{|\boldsymbol{p}|-1} (-1)^{|\boldsymbol{p}|-l+1} \cdot \binom{|\boldsymbol{p}|}{l} \cdot \overline{\text{WFOMC}}(\Psi_{\downarrow |\boldsymbol{p}|-l|}, \boldsymbol{p})$
end for
return $A[\boldsymbol{k}]$

function $\overline{\text{WFOMC}}(\Psi_{\downarrow m}, \boldsymbol{s})$ ▷ Equation (19)
  $S = 0$
  for $\boldsymbol{s}' + \boldsymbol{s}'' = \boldsymbol{s}$ and $|\boldsymbol{s}'| = m$ **do**
    $S \leftarrow S + \text{WFOMC}(\Psi', \boldsymbol{s}') \cdot A[\boldsymbol{s}''] \cdot \prod_{i,j \in [u]} r_{ij}^{s'_i s''_j}$
  end for
  return S
end function

Our results expand and cover previously investigated problems in combinatorics literature, such as counting the number of DAGs with a fixed number of edges and nodes [35]. Such results in the combinatorics literature provide a simple benchmark and sanity-check for our methodology. Hence, we will often check if model-counts for constraints axiomatized using our methodology align with the counts reported in the On-Line Encyclopedia of Integer Sequences (OEIS) [36] — an open-source collection of sequences, enumerating solutions to various counting problems. We illustrate this in the following example.

Example 5. Suppose we want to count the number of DAGs with n labeled nodes and d edges, as can be done through [35]. Clearly, this can be encoded in FOL with DAG constraint as follows

$$\text{FOMC}(Acyclic(R) \wedge |R| = d, n) \tag{23}$$

for an FOL language with only the binary relation R. Indeed, with our implementation of Algorithm 1, the FOMC in (23), for different values of d and n, leads to the same sequence as A081064 of the OEIS [36].

This (and later examples) show that our implementation provides a simple method to enumerate structures respecting interesting combinatorial constraints directly from their logical encoding — without the need to explicitly derive a different formula for each constraint.

5.4. Source and sink

In the following, we show that extending C^2 with a DAG axiom allows us to easily express sources and sinks. We then provide some examples from combinatorics where this is useful.

Definition 7. Let Φ be a first order sentence, possibly containing a binary relation R, a unary relation $Source$ and a unary relation $Sink$. We say that an interpretation ω is a model of $\Phi \wedge Acyclic(R, Source, Sink)$ if:

- ω is a model of $\Phi \wedge Acyclic(R)$,
- the sources of the DAG represented by ω_R are interpreted to be true in ω_{Source},
- the sinks of the DAG represented by ω_R are interpreted to be true in ω_{Sink}.

The $Source$ and the $Sink$ predicates can allow us to encode constraints like $\exists^{=k} x.Source(x)$ or $\exists^{=k} x.Sink(x)$. See the example below.

Theorem 5. *Let $\Psi := \Phi \wedge Acyclic(R, Source, Sink)$, where Φ is a C^2 sentence.* $\text{WFOMC}(\Psi, n)$ *can be computed in polynomial time with respect to the domain cardinality.*

Proof. The sentence Ψ can be equivalently written as:

$$\begin{aligned} &\Phi \wedge Acyclic(R) \\ &\wedge \forall x.Source(x) \leftrightarrow \neg\exists y.R(y, x) \\ &\wedge \forall x.Sink(x) \leftrightarrow \neg\exists y.R(x, y) \end{aligned} \tag{24}$$

which is a C^2 sentence with a DAG constraint, for which Theorem 4 applies. □

Example 6. For an FOL language made only of the binary relation R,

$$\text{FOMC}(Acyclic(R, Source, Sink) \wedge |R| = d \wedge |Source| = 1, n) \tag{25}$$

counts the number of DAGs with n labeled nodes, d edges and a unique source. Similarly,

$$\text{FOMC}(Acyclic(R, Source, Sink) \wedge |Source| = s, n) \tag{26}$$

counts the number of DAGs with n labeled nodes and s sources, while

$$\text{FOMC}(Acyclic(R, Source, Sink) \wedge |Source| = s \wedge |Sink| = s', n) \tag{27}$$

counts the number of DAGs with n labeled nodes, s sources and s' sinks. Indeed, with our axiomatization and implementation, we were able to compute the following sequences from OEIS:

- A350487 (DAGs with n labeled nodes, d edges and 1 source);
- A361718 (DAGs with n labeled nodes and s sources), A003025 (DAGs with n labeled nodes and 1 source), A003026 (DAGs with n labeled nodes and 2 sources);
- A165950 (DAGs with n labeled nodes, 1 source and 1 sink).

These counting problems were also investigated in [23,37].

Example 7. Binary phylogenetic networks [38–41,30] are defined as DAGs consisting only of the following types of nodes:

- one and only one source, which must have outdegree 2;
- leaves, which are sinks with indegree 1;
- tree nodes, which are nodes with indegree 1 and outdegree 2;
- reticulation nodes, which are nodes with indegree 2 and outdegree 1.

Along with their many subclasses, they are commonly used for representing the evolutionary relationships among a group of taxa, taking into account reticulate events like hybridization, horizontal gene transfer and recombination [42]. Counting binary phylogenetic networks is an interesting open problem in combinatorics [38]. Indeed, many recent studies are devoted to providing asymptotic estimates [41,38], or the exact counting when the number of reticulation nodes is limited [38], or estimates and exact counting for related networks, like tree-child and normal networks [43,41,40,31,44,29,39,45,30,46].

Notice that the definition of binary phylogenetic networks is expressible in C^2 with the acyclicity axiom:

$$\begin{aligned} \forall x.[&(Source(x) \wedge \exists^{=2} y.R(x,y)) \\ &\vee (Sink(x) \wedge \exists^{=1} y.R(y,x)) \\ &\vee (\exists^{=1} y.R(y,x) \wedge \exists^{=2} y.R(x,y)) \\ &\vee (\exists^{=2} y.R(y,x) \wedge \exists^{=1} y.R(x,y))] \\ \wedge &|Source| = 1 \wedge Acyclic(R, Source, Sink) \end{aligned} \tag{28}$$

Hence, FOMC with acyclicity axiom can provide a way to count binary phylogenetic networks with a fixed number of nodes in polynomial time. Moreover, this approach is flexible enough to allow to count also other related classes like tree-child networks, and to count binary phylogenetic networks with any constraint expressible in C^2. However, enumerating constraints with counting quantifiers requires an FOMC-preserving reduction to FO^2 with cardinality constraints, with many additional predicates (see [18,11]), and the computational complexity scales super-exponentially with respect to the number of predicates. The number of additional predicates required in this case poses significant scalability issues.

6. WFOMC with connectivity axiom

A *connected graph* is an undirected graph such that for any pair of nodes i and j there exists a path connecting the two nodes. We present a recursive formula for WFOMC of an FO^2 sentence, say Φ, where a special predicate, say R, is axiomatized to be a connected graph, i.e. we count the models $\omega \models \Phi$ such that ω_R represents a connected graph. We begin by revisiting the recursive formula for counting connected graphs on n nodes. Much in the spirit of the previous section, we make observations that allow efficient counting of connected graphs and exploit them for WFOMC with connectivity axiom.

6.1. Counting connected graphs

In a given undirected graph, a *connected component* is a subgraph that is not part of any larger connected subgraph. In a *rooted graph*, one node is labeled in a special way, and called the *root* of the graph. Given a rooted graph, we call its connected component

containing the root as the *rooted-connected component*. We now present a recursive way to count the number c_n of connected graphs on n nodes, as done in [28]. The base case is $c_1 = 1$.

Proposition 5. *The number of rooted graphs on $[n]$ with an m-sized rooted-connected component is given as:*

$$\binom{n}{m} \cdot m \cdot c_m \cdot 2^{\binom{n-m}{2}} \tag{29}$$

where c_m is the number of connected graphs on m nodes.

Proof. Let ω be a rooted graph such that $\omega \downarrow [m]$ forms a rooted-connected component. Since $\omega \downarrow [m]$ is a connected component, there can be no edges between $[m]$ and $[\bar{m}]$. The number of possible connected graphs on $[m]$ is given by c_m. Also, in ω any node in $[m]$ can be chosen to be the root. Hence, the number of ways in which $\omega \downarrow [m]$ can be a rooted-connected component is $m \cdot c_m$. Since $\omega \downarrow [m]$ is a connected-component, there can be no edges between $[m]$ and $[\bar{m}]$, and $\omega \downarrow [\bar{m}]$ can be any $n-m$ sized graph. Hence, $2^{\binom{n-m}{2}}$ subgraphs can be realized on $[\bar{m}]$. Since subgraphs on $[m]$ and $[\bar{m}]$ are realized independently, the total number of graphs on $[n]$ such that $[m]$ is a rooted-connected component is given by $m \cdot c_m \cdot 2^{\binom{n-m}{2}}$. These arguments can be repeated for any rooted graph on $[n]$ with a rooted-connected component of size m. Since there are $\binom{n}{m}$ ways of choosing such subsets, we get formula (29). □

Summing up equation (29) over all m, for $1 \leq m \leq n$, we get the total number of rooted graphs. This is the base of the proof of the following proposition, which gives us a way to recursively count the number of connected graphs on n nodes.

Proposition 6 *([28]). For any m, let c_m be the number of connected graphs on m nodes. Then, the following holds:*

$$c_n = 2^{\binom{n}{2}} - \frac{1}{n} \sum_{m=1}^{n-1} \binom{n}{m} \cdot m \cdot c_m \cdot 2^{\binom{n-m}{2}} \tag{30}$$

Proof. Any rooted graph on $[n]$ has a rooted-connected component of some size m, where $1 \leq m \leq n$. Hence,

$$\sum_{m=1}^{n} \binom{n}{m} \cdot m \cdot c_m \cdot 2^{\binom{n-m}{2}} \tag{31}$$

is counting all the rooted graphs on $[n]$, whose number is $n \cdot 2^{\binom{n}{2}}$. This gives us the following equation, where we have simply rewritten the summation in a different way:

$$n \cdot 2^{\binom{n}{2}} = n \cdot c_n + \sum_{m=1}^{n-1} \binom{n}{m} \cdot m \cdot c_m \cdot 2^{\binom{n-m}{2}} \tag{32}$$

Clearly, this equation can be equivalently rewritten as equation (30). □

6.2. WFOMC with connectivity axiom

Definition 8. Let R be a binary predicate. An interpretation ω is a model of $Connected(R)$ if

- ω_R forms a symmetric and antireflexive relation of R, and
- ω_R forms a connected graph.

Remark 4. Our goal is to compute WFOMC of $\Psi := \forall xy.\Phi(x,y) \wedge Connected(R)$, where $\Phi(x,y)$ is a quantifier-free FO^2 formula interpreted on the domain $[n]$. By definition, this entails that ω_R forms a symmetric and antireflexive relation of R. Hence, we can assume without loss of generality that:

$$\forall xy.\Phi(x,y) \models \forall x.\neg R(x,x) \wedge \forall xy.R(x,y) \rightarrow R(y,x) \tag{33}$$

Definition 9. Let $\Psi := \forall xy.\Phi(x,y) \wedge Connected(R)$, where $\Phi(x,y)$ is a quantifier-free FO^2 formula interpreted on the domain $[n]$, such that (33) holds. For any $m \leq n$, we say that an interpretation ω is a model of $\Psi_{[m]}$, and write $\omega \models \Psi_{[m]}$, if ω is a model of Ψ on $[n]$ and $\omega_R \downarrow [m]$ is a connected component in ω_R.

It can be shown that Definition 9 corresponds to an instantiation of $\Psi_{[m]}$ as introduced in Lemma 2 (see appendix Lemma 4 for a formal proof), where

- $axiom' := Connected(R)$
- $axiom'' := \top$

- $\Theta(x, y) := \neg R(x, y)$

Using this instantiation of $\Psi_{[m]}$ and Lemma 2, we have the following proposition.

Proposition 7. *Let $\Psi := \forall xy.\Phi(x, y) \wedge Connected(R)$ and $\Psi'' := \forall xy.\Phi(x, y)$, where $\Phi(x, y)$ is a quantifier-free FO^2 formula such that (33) holds. Then:*

$$\text{WFOMC}(\Psi_{[m]}, \boldsymbol{k}) = \sum_{\substack{\boldsymbol{k'}+\boldsymbol{k''}=\boldsymbol{k} \\ |\boldsymbol{k'}|=m}} \text{WFOMC}(\Psi, \boldsymbol{k'})\text{WFOMC}(\Psi'', \boldsymbol{k''}) \prod_{i,j\in[u]} r_{ij}^{k'_i k''_j} \tag{34}$$

where $r_{ij} := \sum_l n_{ijl} v_l$, and n_{ijl} is 1 if $ijl(x, y) \models \Phi(\{x, y\}) \wedge \neg R(x, y)$ and 0 otherwise.

Similar to Proposition 5, we will now compute the WFOMC for interpretations with rooted connected-components of size m.

Proposition 8. *Let $\Psi := \forall xy.\Phi(x, y) \wedge Connected(R)$ be a sentence interpreted over $[n]$, where $\Phi(x, y)$ is a quantifier-free FO^2 formula such that (33) holds. Then the WFOMC of all the models $\omega \models \forall xy.\Phi(x, y)$ with a rooted-connected component of size m is given as:*

$$\binom{n}{m} \cdot m \cdot \text{WFOMC}(\Psi_{[m]}, \boldsymbol{k}) \tag{35}$$

Proof. The proof idea is similar to the one of Proposition 5. There are $\binom{n}{m}$ ways of choosing an m-sized subset C in $[n]$. Given such a set C, the number of models $\omega \models \forall xy.\Phi(x, y)$ with C as a connected component w.r.t R is $\text{WFOMC}(\Psi_{[m]}, \boldsymbol{k})$. Finally, if we also allow a node in C to be distinguished as a root, then we have m ways of choosing the root. □

Summing up equation (35) over all m, for $1 \leq m \leq n$, we get the total number of interpretations $\omega \models \forall xy.\Phi(x, y)$, where ω_R is a rooted graph. This is the base of the proof of the following proposition, which gives us a way to recursively compute the WFOMC of $\Psi := \forall xy.\Phi(x, y) \wedge Connected(R)$.

Proposition 9. *Let $\Psi := \forall xy.\Phi(x, y) \wedge Connected(R)$ and $\Psi'' := \forall xy.\Phi(x, y)$, where $\Phi(x, y)$ is a quantifier-free FO^2 formula such that (33) holds. Then:*

$$\text{WFOMC}(\Psi, \boldsymbol{k}) = \text{WFOMC}(\Psi'', \boldsymbol{k}) - \frac{1}{n} \sum_{m=1}^{n-1} \binom{n}{m} \cdot m \cdot \text{WFOMC}(\Psi_{[m]}, \boldsymbol{k}) \tag{36}$$

Proof. The proof idea is similar to the one of Proposition 6. We notice that

$$\sum_{m=1}^{n} \binom{n}{m} \cdot m \cdot \text{WFOMC}(\Psi_{[m]}, \boldsymbol{k}) \tag{37}$$

sums the WFOMC of all the models ω of $\forall xy.\Phi(x, y)$ for which ω_R is a simple graph with an R-rooted-connected component of size m, where $1 \leq m \leq n$. But any rooted graph has a rooted-connected component of some size m, where $1 \leq m \leq n$. Hence, equation (37) computes the weighted sum of all models of $\forall xy.\Phi(x, y)$ where ω_R is a rooted graph. This is equal to n times the WFOMC of $\forall xy.\Phi(x, y)$, because we have n choices for assigning a root in each model. Thus, we get the following equation

$$\begin{aligned} &n \cdot \text{WFOMC}(\forall xy.\Phi(x, y), \boldsymbol{k}) = \\ &n \cdot \text{WFOMC}(\Psi_{[n]}, \boldsymbol{k}) + \sum_{m=1}^{n-1} \binom{n}{m} \cdot m \cdot \text{WFOMC}(\Psi_{[m]}, \boldsymbol{k}) \end{aligned} \tag{38}$$

and since $\Psi_{[n]}$ is equivalent to Ψ, it can be rewritten as equation (36). □

Using Proposition 7 and Proposition 9 we can derive a recursive algorithm, very similar to Algorithm 2, for computing $\text{WFOMC}(\Psi, \boldsymbol{k})$. This can be achieved by replacing line 6 and line 13 of Algorithm 2 with the formulas coming from equations (34) and (36). That is, replacing line 6 and line 13, with the following operations:

6: $A[\boldsymbol{p}] \leftarrow \text{WFOMC}(\Psi'', \boldsymbol{p}) - \frac{1}{|\boldsymbol{p}|} \sum_{m=1}^{|\boldsymbol{p}|-1} \binom{|\boldsymbol{p}|}{m} \cdot m \cdot \overline{\text{WFOMC}}(\Psi_{[m]}, \boldsymbol{p})$

and

13: $S \leftarrow S + A[\boldsymbol{s'}] \cdot \text{WFOMC}(\Psi'', \boldsymbol{s''}) \cdot \prod_{i,j\in[u]} r_{ij}^{s'_i s''_j}$

The analysis of the modified algorithm for WFOMC with a connectivity constraint (see Algorithm 3) is provided in the appendix. Almost all of the tractability analysis for Algorithm 2 transfers identically to Algorithm 3, showing that the algorithm is tractable. Hence giving us the following theorem:

Theorem 6. *Let $\Psi := \Phi \wedge Connected(R)$, where Φ is a C^2 formula. Then* WFOMC(Ψ, n) *can be computed in polynomial time with respect to the domain cardinality.*

The following examples show possible applications of our results in the realm of combinatorics.

Example 8. For a FOL language made only of the binary relation R,

$$\text{FOMC}(Connected(R) \wedge |R| = 2d, n) \tag{39}$$

counts the number of connected graphs with n labeled nodes and d edges. For it to be non-zero, d must be between $n-1$ (trees) and $\binom{n}{2}$ (complete graph). Such counting problem is studied in chapter 6.3 of [47]. With our implementation, we were able to get the array A062734 of OEIS, which is also equivalent to A123527 and A343088. Its diagonals also give the following sequences: A000272 (number of trees on n nodes), A057500 (number of connected unicyclic graphs on n nodes), A061540, A061541, A061542, A061543, A096117, A061544 A096150, A096224, A182294, A182295, A182371.

Example 9. Suppose we want to count the number of connected graphs with n labeled nodes, each one colored by one of three colors, with the condition that adjacent nodes must have different colors. This problem was already studied and solved in [48], where such number is denoted by $m_n(3)$. We consider a FOL language with a relational symbol $R/2$ for representing the connected graph, and three relational symbols $A/1, B/1, C/1$ representing the colors. The fact that one and only one color is assigned to each node can be encoded with the sentence

$$\begin{aligned} \Psi_1 := \forall x.(A(x) \vee B(x) \vee C(x)) \\ \wedge \neg(A(x) \wedge B(x)) \wedge \neg(A(x) \wedge C(x)) \wedge \neg(B(x) \wedge C(x)) \end{aligned} \tag{40}$$

while the fact that adjacent nodes must have different colors can be encoded with the sentence

$$\begin{aligned} \Psi_2 := \forall xy.(A(x) \wedge R(x,y) \rightarrow \neg A(y)) \\ \wedge (B(x) \wedge R(x,y) \rightarrow \neg B(y)) \\ \wedge (C(x) \wedge R(x,y) \rightarrow \neg C(y)) \end{aligned} \tag{41}$$

Hence, the number of colored graphs that we want can be computed as

$$\text{FOMC}(\Psi_1 \wedge \Psi_2 \wedge Connected(R), n) \tag{42}$$

and with our implementation we were able to compute sequence A002028 of OEIS.

7. Trees and forests

In this section, we investigate the domain liftability of C^2 where one of the relational symbols is axiomatized to be a *tree* or a *forest*. Tree axioms have been previously investigated in [20]. However, we provide a completely different method of WFOMC for both directed and undirected tree axioms. Furthermore, we also demonstrate domain liftability of both directed and undirected forest axioms — an open problem raised in the conclusion of [20].

7.1. WFOMC with directed tree and directed forest axioms

Directed trees are DAGs with exactly one node with indegree zero and with all the other nodes having indegree 1. Similarly, *directed forests* are DAGs such that every node has indegree at most 1. (Equivalently, directed forests are DAGs such that each connected component is a directed tree.)

Definition 10. An interpretation ω is a model of $DirectedTree(R, Root)$ if ω_R forms a directed tree whose root (i.e. the node with no R-parent) is the unique domain element interpreted to be true in ω_{Root}.

Definition 11. An interpretation ω is a model of $DirectedForest(R)$ if ω_R forms a directed forest.

Using the definition of directed trees, as DAGs with exactly one root and with all the other nodes having indegree 1, we get the following proposition.

Proposition 10. *$\Phi \wedge DirectedTree(R, Root)$ is equivalent to*

$$\Phi \wedge Acyclic(R) \wedge |Root| = 1 \wedge \forall x. \neg Root(x) \rightarrow \exists^{=1} y. R(y, x) \tag{43}$$

for any sentence Φ. Hence, WFOMC with directed tree axiom can be modularly reduced to WFOMC with DAG axiom.

Proof. Let ω be a model of formula (43), so that ω_R is a DAG. Since $\omega \models \forall x.\neg Root(x) \rightarrow \exists^{=1} y.R(y,x)$, we have that any element c such that $\omega \models \neg Root(c)$, has exactly one R-parent. Since $\omega \models (|Root| = 1)$, we have that ω_R is a DAG with only one node, say r, such that r does not have exactly one R-parent, and all other nodes have exactly one R-parent. But ω_R is a DAG, hence it necessarily has one node with no parent, which hast to be r. Hence, ω_R is a directed rooted tree. It can also be checked that any $\omega \models \Phi$ such that ω_R is directed rooted tree is a model of formula (43). □

Analogously, using the definition of directed forests, as DAGs where each node has indegree at most 1, we get the following proposition:

Proposition 11. $\Phi \wedge DirectedForest(R)$ *is equivalent to*

$$\Phi \wedge Acyclic(R) \wedge \forall x.\exists^{<=1} y.R(y,x)$$

for any sentence Φ. *Hence, WFOMC with directed forest axiom can be modularly reduced to WFOMC with DAG axiom.*

Since both $DirectedTree(R, Root)$ and $DirectedForest(R)$ can be modularly reduced to WFOMC with DAG axiom and additional C^2 constraints, the domain liftability results for the DAG axiom — such as Theorem 4 — also transfer to $DirectedTree(R, Root)$ and $DirectedForest(R)$ axioms.

Remark 5. In the definition above we used the predicate $Root$ only for directed trees. Notice, however, that we could define it also for directed forests, by imposing the satisfaction of the following sentence

$$\forall x.Root(x) \leftrightarrow \neg\exists y.R(y,x)$$

In a similar way, one could define a unary predicate $Leaf$ for representing leaves, both for directed trees and for directed forests, using the following sentence:

$$\forall x.Leaf(x) \leftrightarrow \neg\exists y.R(x,y)$$

7.2. WFOMC with tree axiom

We will now focus on the undirected tree axiom.

Definition 12. Let R be a binary predicate, an interpretation ω is a model of $Tree(R)$ if

- ω_R forms a symmetric and antireflexive relation of R, and
- ω_R forms a Tree

Remark 6. Notice that one could also define a unary predicate $Leaf$ for representing leaves by imposing the satisfaction of the following sentence:

$$\forall x.Leaf(x) \leftrightarrow \exists^{=1} y.R(x,y)$$

This applies also to the case of undirected forests, which will be introduced later.

Note that a connected graph on n nodes is a tree if and only if it has $n-1$ edges [49, Theorem 2.1]. This gives us the following proposition:

Proposition 12. *For any sentence* Φ *on a domain of cardinality n, we have that* $\Phi \wedge Tree(R)$ *is equivalent to*

$$\Phi \wedge Connected(R) \wedge |R| = 2n - 2$$

Hence, WFOMC with tree axiom can be modularly reduced to WFOMC with connectivity axiom.

Remark 7. Since $Tree(R)$ can be modularly reduced to WFOMC with connectivity axiom and additional cardinality constraints, the domain liftability results for the connectivity axiom — such as Theorem 6 — also transfer to the $Tree(R)$ axiom. These results were already established in [20], but we are providing a different way to arrive at them.

7.3. WFOMC with forest axiom

We now present an algorithm for extending domain liftability of C^2 with an undirected forest axiom. We first prove the formula for counting undirected forests as presented in [25]. This will then form the basis of our algorithm for WFOMC with forest axiom, which is subsequently presented.

Proposition 13. *Let t_m denote the number of trees on m labeled nodes, and let f_n be the number of forests on n labeled nodes. Then,*

$$f_n = \sum_{m=1}^{n} \binom{n-1}{m-1} t_m f_{n-m} \tag{44}$$

Proof. Without loss of generality, we can assume the n labeled nodes to be the set $[n]$. We will prove the proposition by observing that any forest on such nodes can be viewed as a tree containing the node 1 plus a forest made up of the remaining connected components. Consider an arbitrary forest on $[n]$, and let m denote the number of nodes of the connected component containing the node 1. Clearly, for an arbitrary forest, m can be any number from 1 to n. Now, fixing m, the number of different trees containing node 1 and $m-1$ other nodes is given by $\binom{n-1}{m-1} t_m$, where $\binom{n-1}{m-1}$ represents the number of ways to choose $m-1$ nodes from $[\bar{1}]$, and t_m denotes the number of different trees on the m selected nodes (1 included). On the other hand, there are f_{n-m} different forests on the remaining $n-m$ nodes, independently of the tree containing 1. It follows that the number of forests such that the connected component containing 1 has $m-1$ other nodes is $\binom{n-1}{m-1} t_m f_{n-m}$. To count all possible forests on n nodes, we only need to sum up the contributions from each m, and this gives us the desired recursive formula. □

Note that the number t_m of trees on m nodes is given by m^{m-2} (Cayley's formula [25]). However, this is irrelevant for what follows.

Definition 13. Let R be a binary predicate. An interpretation ω is a model of $Forest(R)$ if ω_R forms a forest.

Remark 8. As done before for connectivity, since $\forall xy.\Phi(x,y) \wedge Forest(R)$ assumes that ω_R forms a symmetric and anti-reflexive relation, we can assume without loss of generality that

$$\forall xy.\Phi(x,y) \models \forall x.\neg R(x,x) \wedge \forall xy.R(x,y) \rightarrow R(y,x) \tag{45}$$

Definition 14. Let $\Psi := \forall xy.\Phi(x,y) \wedge Forest(R)$, where $\Phi(x,y)$ is a quantifier-free FO^2 formula interpreted on the domain $[n]$, such that (45) holds. For any $m \leq n$, we say that an interpretation ω is a model of $\Psi_{[m]}$, and write $\omega \models \Psi_{[m]}$, if ω is a model of Ψ on $[n]$ and $\omega_R \downarrow [m]$ forms a connected component of ω_R (and hence a tree).

It is easy to check that $\Psi_{[m]}$ in Definition 14 is the same as $\Psi_{[m]}$ in Lemma 2, where we instantiate $axiom'$, $axiom''$ and $\Theta(x,y)$ as follows:

- $axiom' := Tree(R)$
- $axiom'' := Forest(R)$
- $\Theta(x,y) := \neg R(x,y)$

A formal proof can be found in the appendix, as Lemma 5. Using this characterization and Lemma 2, we have the following proposition.

Proposition 14. *Let $\Psi := \forall xy.\Phi(x,y) \wedge Forest(R)$, and $\Psi' := \forall xy.\Phi(x,y) \wedge Tree(R)$, where $\Phi(x,y)$ is a quantifier free FO^2 formula satisfying the condition in Remark 8. Then:*

$$\text{WFOMC}(\Psi_{[m]}, \boldsymbol{k}) = \sum_{\substack{\boldsymbol{k}=\boldsymbol{k}'+\boldsymbol{k}'' \\ |\boldsymbol{k}'|=m}} \prod_{i,j} r_{ij}^{k_i' k_j''} \text{WFOMC}(\Psi', \boldsymbol{k}') \text{WFOMC}(\Psi, \boldsymbol{k}'') \tag{46}$$

where $r_{ij} := \sum_l n_{ijl} v_l$, and n_{ijl} is 1 if $ijl(x,y) \models \Phi(\{x,y\}) \wedge \neg R(x,y)$ and 0 otherwise.

Proposition 15. *Let $\Psi := \forall xy.\Phi(x,y) \wedge Forest(R)$ be interpreted over $[n]$, where $\Phi(x,y)$ is a quantifier free FO^2 formula satisfying the condition in Remark 8. Then:*

$$\text{WFOMC}(\Psi, \boldsymbol{k}) = \sum_{m=1}^{n} \binom{n-1}{m-1} \text{WFOMC}(\Psi_{[m]}, \boldsymbol{k}) \tag{47}$$

Proof. The proof idea is essentially the same as the one for Proposition 13. Let Ψ_C be the generalization of $\Psi_{[m]}$ to a generic subset $C \subseteq [n]$. Clearly, ω is a model of Ψ if and only if ω is a model of Ψ_C for a subset C containing the node 1. Notice also that for any ω there can be only one subset C such that $\omega_R \downarrow C$ is the connected component of ω_R containing the node 1. Hence,

$$\text{WFOMC}(\Psi, \boldsymbol{k}) = \sum_{\substack{C \subseteq [n] \\ 1 \in C}} \text{WFOMC}(\Psi_C, \boldsymbol{k}) \tag{48}$$

Now, $\text{WFOMC}(\Psi_{[m]}, \boldsymbol{k})$ is the same as $\text{WFOMC}(\Psi_C, \boldsymbol{k})$ for any $C \subseteq [n]$ with $|C| = m$. Furthermore, there are $\binom{n-1}{m-1}$ ways of choosing a subset C of $[n]$ in such a way that it contains 1. This gives us equation (47). □

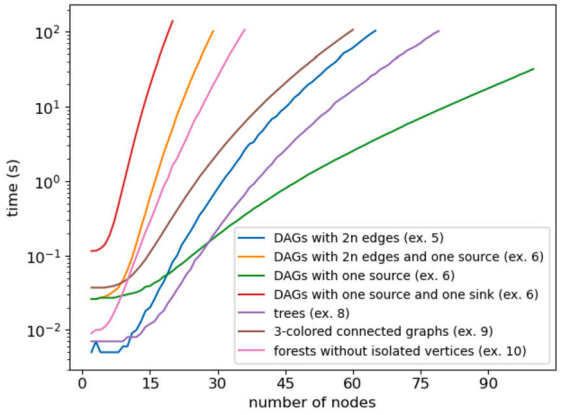


Fig. 1. Run-times of all experiments as function of the domain cardinality n. For the WFOMC encoding see the referred examples. The number of active 1-types for each formula, and if cardinality constraints on binary predicates were used, is reported in Table 2 of the appendix.

We provide the pseudocode for computing $\text{WFOMC}(\Psi, \boldsymbol{k})$ as Algorithm 4 in the appendix. The only changes w.r.t Algorithm 2 are that line 6 is substituted with

6: $A[\boldsymbol{p}] \leftarrow \sum_{m=1}^{|\boldsymbol{p}|} \binom{|\boldsymbol{p}|-1}{m-1} \overline{\text{WFOMC}}(\Psi_{[m]}, \boldsymbol{p})$

and that $\text{WFOMC}(\Psi', \boldsymbol{k}')$ in line 13 is using WFOMC with tree axiom, whose liftability is ensured by Remark 7. With this in mind, it can be proved that the algorithm runs in polynomial time w.r.t domain cardinality in the same way of Algorithm 2. Thus, we can derive the domain liftability of C^2 with the forest axiom by using the modular reductions from [5] and [18] (in the same way as we did for DAG and connected axioms).

Theorem 7. *Let $\Psi := \Phi \wedge Forest(R)$, where Φ is a C^2 formula. Then $\text{WFOMC}(\Psi, n)$ can be computed in polynomial time with respect to the domain cardinality.*

Again, we provide an example to show that our results can be useful in enumerative combinatorics.

Example 10. The number of forests without isolated vertices on n labeled nodes can be computed as

$$\text{FOMC}(\forall x.\exists y.R(x,y) \wedge Forest(R), n) \tag{49}$$

for a FOL language made only of the binary relation R. Using the procedure in [5], this is reduced to a form to which Algorithm 4 can be directly applied:

$$\text{WFOMC}(\forall xy.S(x) \vee \neg R(x,y) \wedge Forest(R), n) \tag{50}$$

for a FOL language made only of $R/2$ and $S/1$, with S having weights $w(S) = 1$ and $\bar{w}(S) = -1$, and R having weights $w(R) = \bar{w}(R) = 1$. In fact, with our implementation, we were able to compute sequence A105784 of OEIS.

8. Experiments

In this section we provide an empirical analysis of the practical efficiency and expressivity of the constraints introduced in this paper. We implemented our WFOMC algorithms (Algorithms 2, 3, and 4) in Python,[1] and the experiments on an Apple M3 Pro computer. The implementation of the cardinality constraints on unary predicates was done by filtering the 1-type cardinality vectors $\boldsymbol{k}$, as in [11]; while for binary predicates we used symbolic weights,[2] as in [13].

8.1. Combinatorial examples

We experiment with the combinatorial examples provided in the previous sections, and run-times are reported in Fig. 1. Although our approach provides a general language for easily formulating complex combinatorial problems, and our WFOMC algorithms run in polynomial time w.r.t domain cardinality, our experiments show that they are far from optimal. This is especially pronounced in the run-time for computing the number of trees, which admits the simple closed-form formula n^{n-2}. In our observation, the number of

[1] Code is available at https://github.com/dbizzaro/WFOMC-beyond-FOL.

[2] The use of symbolic weights is equivalent to the procedure in the proof of Theorem 2, as explained in the appendix (to Section 3).

(active) 1-types[3] and the presence of cardinality constraints on binary predicates constitute key sources of intractability (see Appendix Table 2). As the former leads to more and larger operations, the latter leads to a complex polynomial interpolation task. Hence, the generality of our proposed framework comes potentially at the cost of lower efficiency, in comparison to results obtained by more fine-grained combinatorics. These results show that more work needs to be done on better practical implementation of the proposed WFOMC results, potentially integrating solutions like [50,51].

8.2. Markov logic networks

A *Markov Logic Network (MLN)* [3] is a set of weighted formulas $\Phi := \{w_i : \phi_i\}_i$, where each ϕ_i is a quantifier free FOL formula with weight $w_i \in \mathbb{R} \cup \{\infty\}$. The formulas with weight ∞ are hard constraints, i.e., any world that does not follow those constraints has probability 0. Let $\Phi_{\mathbb{R}}$ be the subset of real-valued weighted formulas in Φ, and let Φ_∞ be the hard constraints. Given a domain Δ, an MLN Φ defines a probability distribution over the set of possible interpretations:

$$P_\Phi^\Delta(\omega) = \frac{\mathbb{1}_{\omega \models \Phi_\infty}}{Z_\Phi^\Delta} \exp\left(\sum_{w_i : \phi_i \in \Phi_{\mathbb{R}}} w_i \cdot n(\phi_i, \omega) \right) \tag{51}$$

where

- $n(\phi_i, \omega)$ represents the number of true groundings of ϕ_i in ω;
- $\mathbb{1}_{\omega \models \Phi_\infty}$ is 1 when $\omega \models \Phi_\infty$ and 0 otherwise;
- Z_Φ^Δ is a normalization constant (ensuring that P_Φ^Δ is a probability distribution) called *partition function*.

One of the applications of WFOMC is probabilistic inference in MLNs [5]. For every finitely-weighted formula $w_i : \phi_i$ in an MLN Φ, we introduce a fresh predicate P_i/a_i, whose arity a_i is the number of free variables in ϕ_i. We define a weight function $(w, \bar{w})$ such that $w(P_i) = \exp(w_i)$ and $\bar{w}(P_i) = 1$ for each i, while $w(R) = \bar{w}(R) = 1$ for every other predicate R. Moreover, let Ψ denote the conjunction of Φ_∞ with all the formulas

$$\forall x_1 \dots x_{a_i}.\ P_i(x_1, \dots, x_{a_i}) \leftrightarrow \phi(x_1, \dots, x_{a_i}) \tag{52}$$

Then, for any query sentence ϕ, probabilistic inference can be computed in the following way:

$$P_\Phi^\Delta(\phi) = \frac{\text{WFOMC}(\Psi \wedge \phi, |\Delta|)}{\text{WFOMC}(\Psi, |\Delta|)} \tag{53}$$

This means that inference can be performed in polynomial time (w.r.t domain cardinality) whenever Ψ and ϕ are domain-liftable.

The constrains introduced in this paper, i.e., $DAG(R)$, $Connected(R)$ and $Forest(R)$, can also be used as hard constraints (Ψ) or as queries (ϕ), while admitting polynomial time inference in MLNs. The experiments below demonstrate that the probability distributions that can be obtained by incorporating these constraints in an MLN can be difficult to achieve otherwise. Such distributions may result in a better model whenever we have reasons to believe that the graph formed by a particular binary relation should be a DAG, a connected graph or a forest. All the distributions in the experiments below are computed (exactly) using the algorithms introduced in the paper.

8.3. Graph statistics

Let us begin by examining the simplest scenario: an MLN that models only a directed (resp. undirected) graph. When restricting the MLN to consider only graphs that are DAGs (resp. connected, or forests), then all the statistics about the graphs can be different. An example of this is shown in Fig. 2. In this experiment, we show that MLNs with our constraints lead to different edge distributions compared to the unconstrained cases. For this analysis, we considered a fixed prior on the distribution of edges given by the weighted formula $-1 : R(x, y)$ (which encodes a preference for sparse graphs), and plotted the distributions of the number of edges. We compared the distributions when the graph constraints are imposed (blue curves), against what are arguably their best approximations in FO^2 (orange curves). For connectivity and forests, we compared also the distribution produced by the corresponding FO^2 formula together with the cardinality constraint that matches the support of the number of edges for the two cases[4] (green curves). The actual distributions of the edge count for domain size $n = 20$ is available in the appendix (Fig. 5). As expected, the DAG and forest constraints favor more sparsity, while the opposite is true for connectivity. The next example will show that even when this particular effect is not very pronounced, the overall effect of a global constraint like connectivity can be significant (even compared to cardinality constraints).

[3] Active 1-types for a formula are the 1-types that are consistent with the formula.

[4] Connected graphs can have any number of edges between $n-1$ and $\binom{n}{2}$, while forests can have any number of edges between 0 and $n-1$.

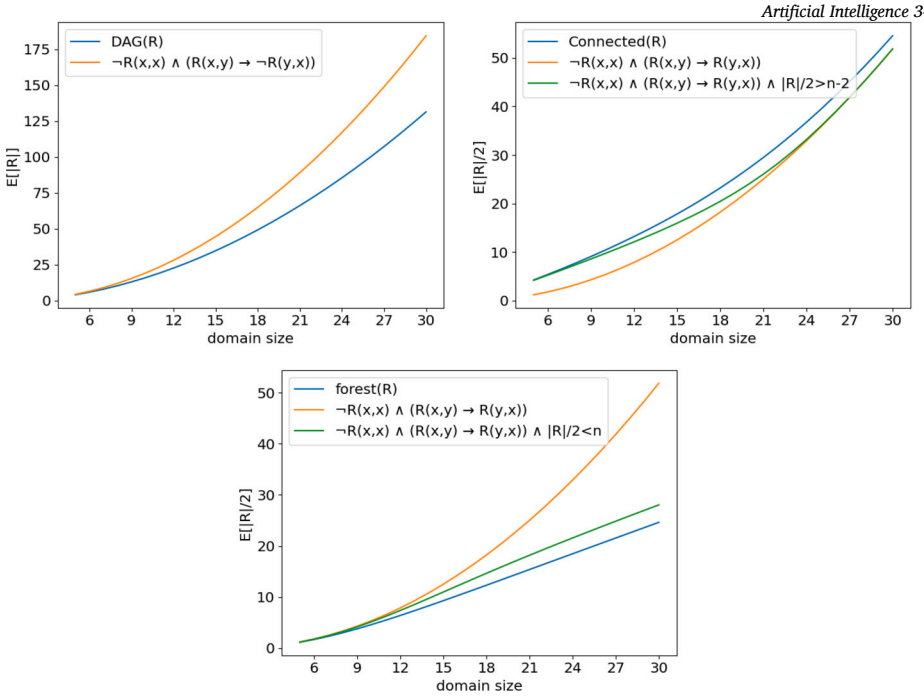


Fig. 2. Expected values of the number of edges of the directed/undirected graphs produced by MLNs with only the predicate $R/2$. The legends report the hard constraints, while the soft constraint is always $-1 : R(x, y)$.

8.4. Smokers & friends with connectivity constraint

We will now analyze the widely investigated social network example of "smokers and friends" in the sparse regime. The example is encoded with the following soft constraints:

$$w_S : S(x) \tag{54}$$

$$w_F : F(x, y) \tag{55}$$

$$w_P : S(x) \wedge F(x, y) \rightarrow S(y) \tag{56}$$

where $S/1$ is the predicate for smokers and $F/2$ for friendship. If the weight w_P is positive, then the MLN models the fact that friends of smokers are more likely to be smokers. The network can be made sparse by setting the weight w_F to be negative — larger negative weight leads to more sparsity.

The partition function for this MLN can be computed (in polynomial time) as the WFOMC of the following formula:

$$\forall xy.\ P(x, y) \leftrightarrow (S(x) \wedge F(x, y) \rightarrow S(y)) \tag{57}$$

with the following weight function:

$$w(S) = \exp(w_S), \quad \bar{w}(S) = 1 \tag{58}$$

$$w(F) = \exp(w_F), \quad \bar{w}(F) = 1 \tag{59}$$

$$w(P) = \exp(w_P), \quad \bar{w}(P) = 1 \tag{60}$$

Fig. 3 compares the distributions of the number of smokers for the following three cases[5]:

- the graph realized by F must be a connected undirected graph (blue);
- the graph realized by F can be any undirected graph (orange);
- the graph realized by F must be an undirected graph with at least as many edges as the number of nodes minus one (which is the minimum number of edges for connected graphs; green).

[5] Each of the three cases is modeled by adding the corresponding hard constraint to the WFOMC of equation (57). The hard constraints are formally expressed in the legends of Fig. 3.

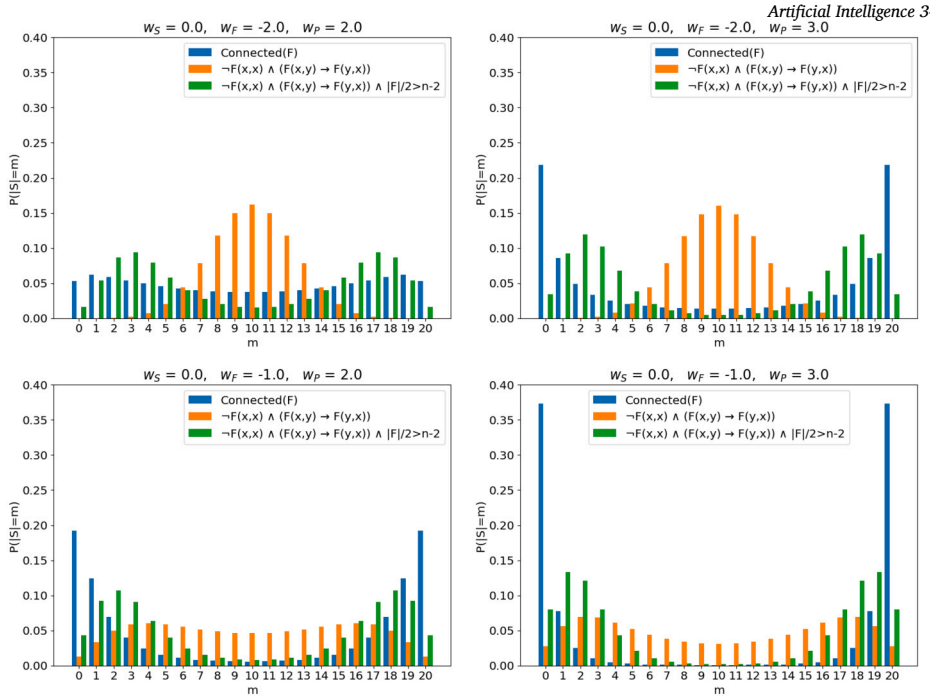


Fig. 3. Distributions of smokers for different weights. The legends report the hard constraints, while the titles report the weights of the soft constraints. The domain size is fixed to $n = 20$.

In this experiment (Fig. 3), we analyze four different values of weight combinations for w_F and w_P, while w_S is fixed to be zero. We can see that the distribution becomes more concentrated at the extremes (either all smokers or none) when: (i) increasing the bias towards the satisfaction of the "smoker and friends" formula (i.e. increasing w_P), or (ii) decreasing the sparsity of the friendship network (i.e. increasing w_F). This aligns with our intuition: (i) as w_P increases, any connected component of the friendship network is more likely to have either all smokers or none; (ii) the less sparsity (i.e., the larger w_F), the more likely to have one large connected component comprising most of the nodes — and all the nodes of such component tend to become homogeneously smokers or non-smokers, modulated by the weight w_P. The interplay of w_F and w_P controls the connectivity and smoking-homogeneity of the networks modeled by the MLN.

Given that connectivity controls the "spreading" of homogeneity of smoking, adding connectivity constraints allow us to further modulate the global distribution of smokers in the network — as reflected in the blue histograms in Fig. 3. Note that the connected networks still maintain high sparsity, as evidenced in Fig. 2, and shown more precisely in the appendix (Fig. 6, top-right). Importantly, connectivity is forcing the graph to have only one connected component, so smoking behavior can "spread" across the entire graph, making most of the nodes either smokers or non-smokers. As expected, the distribution of smokers under the connectivity constraint is consistently more concentrated at the extremes than in the other two cases.

Analogous experiments with DAG and forest constraints instead of connectivity are reported in the appendix (Figs. 7 and 8). In these cases, the differences are mainly driven by the corresponding increase in sparsity. Although these constraints demonstrate different artifacts than FOL and connectivity, we leave there qualitative analysis to future work for more relevant examples. For instance, citation networks for DAGs, and genealogy networks for forests.

Finally, Fig. 4 shows the run-times of computing the partition function of the "smokers and friends" MLNs, i.e. the run-times of the WFOMC of each of the expressions in the legend, together with formula (57). To avoid numerical issues, the weights were represented as fractions, and exact calculations were performed.

9. Conclusion

We investigate the domain-liftability of the first order logic fragment with two variables and counting quantifiers (C^2), with additional graph theoretical constraints. We show that the domain liftability of C^2 is preserved when one of the relations in the language is restricted to represent an acyclic graph, a connected graph, a forest (resp. a directed forest), or a tree (resp. a directed tree). A key novel idea used consistently throughout our work is *counting by splitting*. The generality of this principle can potentially aid the lifting of many other novel constraints. Besides their application in statistical relational learning, our results provide a uniform framework for combinatorics on different types of multi-relational graphs. In fact, our work covers and extends a vast array of results in combinatorics, such as counting phylogenetic networks and enumerating acyclic graphs/forests with different constraints. Our

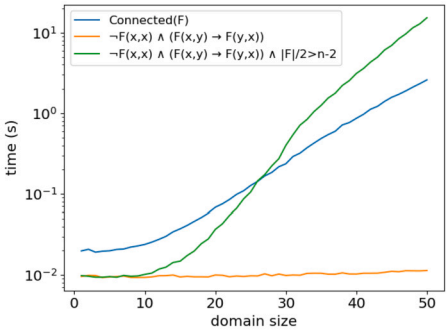


Fig. 4. Run-times of computing the partition function of the "smokers and friends" MLNs.

work (along with [20]) can be seen as extending the relational language for constraints that admit lifted inference beyond first order logic. Hence, these results motivate the following open problem:

Is there a formal language that captures all tractable counting problems for labeled relational structures?

Our experimental analysis shows that the new tractable constrains lead to potentially useful differences in distributions expressed by Markov Logic Networks. However, we observe that our algorithms, though polynomial in time, have scalability issues, and often do not lead to the optimal computational complexity. This motivates further research into designing more efficient practical implementations, and in understanding the gap between domain lifted and optimal inference.

CRediT authorship contribution statement

Sagar Malhotra: Writing – review & editing, Writing – original draft, Visualization, Supervision, Project administration, Methodology, Formal analysis, Conceptualization. **Davide Bizzaro:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Luciano Serafini:** Writing – review & editing, Supervision, Project administration, Investigation, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Appendix to Section 3

Theorem 2. *([18], slightly reformulated) Let Φ be a first-order logic sentence and let Γ be an arbitrary cardinality constraint. Then,* $\text{WFOMC}(\Phi \wedge \Gamma, \boldsymbol{k})$ *can be computed in polynomial time with respect to the domain cardinality, relative to the* $\text{WFOMC}(\Phi, \boldsymbol{k})$ *oracle.*

Proof. Let us consider an FOL language $\mathcal{L}$ that contains r relational symbols denoted by $\{R_i/a_i\}_{i\in[r]}$. Let ω be an interpretation and let $\boldsymbol{\mu} = \langle |R_1|, \ldots, |R_r| \rangle$ be the vector comprising the cardinality of each predicate R_i in ω. Now, $\text{W}(\omega)$ can be evaluated using the definition of symmetric weight functions (Definition 2). Any two interpretations that have the same predicate cardinalities $\boldsymbol{\mu}$ as ω have the same weight $\text{W}(\omega)$. Therefore, we use $\text{W}_{\boldsymbol{\mu}}$ to indicate the weight $\text{W}(\omega)$, where $\omega \vDash \boldsymbol{\mu}$. Given an FOL formula Φ, let $A_{\boldsymbol{\mu}}$ be the number of interpretations $\omega \vDash \Phi \wedge \boldsymbol{\mu}$, then the following holds:

$$\text{WFOMC}(\Phi, \boldsymbol{k}) = \sum_{\boldsymbol{\mu}} A_{\mu} \text{W}_{\boldsymbol{\mu}} \tag{61}$$

For each predicate R_i/a_i in the FOL language $\mathcal{L}$, there exist n^{a_i} ground atoms. Therefore, there are $n^{\sum_{i\in[r]} a_i}$ potential values of $\boldsymbol{\mu}$, which means that there are polynomially many vectors $\boldsymbol{\mu}$ with respect to n. By evaluating $\text{WFOMC}(\Phi, \boldsymbol{k})$ for $n^{\sum_{i\in[r]} a_i}$ distinct weight function pairs $(w, \bar{w})$, we can obtain a non-singular linear system of $n^{\sum_{i\in[r]} a_i}$ equations on the $n^{\sum_{i\in[r]} a_i}$ variables $A_{\boldsymbol{\mu}}$. This system can be solved using Gauss-elimination algorithm in $O(n^{3\sum_{i\in[r]} a_i})$ time. Then, having found the numbers $A_{\boldsymbol{\mu}}$, we can compute the value of any cardinality constraint as follows:

$$\text{WFOMC}(\Phi \wedge \Gamma, \boldsymbol{k}) = \sum_{\boldsymbol{\mu} \vDash \boldsymbol{\Gamma}} A_{\mu} \text{W}_{\boldsymbol{\mu}} \tag{62}$$

where $\boldsymbol{\mu} \vDash \boldsymbol{\Gamma}$ represents the fact that the predicate cardinalities $\boldsymbol{\mu}$ satisfy the cardinality constraint Γ. Since, there is only a polynomial number of vectors $\boldsymbol{\mu}$, equation (62) can be computed in polynomial time. □

Remark 9. In equation (62) we assume that $\boldsymbol{\mu} \models \Gamma$ can be checked in polynomial time w.r.t. n. This is a reasonable assumption for all our purposes.

A.1. Practical implementation of cardinality constraints

Note that the proof of Theorem 2 assumes only a black-box access to the WFOMC$(\Phi, \boldsymbol{k})$ oracle. However, for all the algorithms introduced in the paper we can treat WFOMC$(\Phi, \boldsymbol{k})$ as a polynomial with symbolic weights for each predicate as variables. For example, let $\Phi := \forall xy.\neg R(x,x) \wedge (R(x,y) \rightarrow R(y,x))$ (encoding for an undirected graph without self-loops). Then, equation (8) becomes:

$$\text{WFOMC}(\Phi, (w, \bar{w}), n) = (w(R)^2 + \bar{w}(R)^2)^{\binom{n}{2}} = \sum_{k=1}^{\binom{n}{2}} \binom{\binom{n}{2}}{k} w(R)^{2k} \bar{w}(R)^{2(\binom{n}{2}-k)}$$

which is a polynomial in the variables $w(R)$ and $\bar{w}(R)$. Similarly to equation (61), we have that this polynomial can be written as $\sum_{\boldsymbol{\mu}} A_{\boldsymbol{\mu}} \mathrm{w}_{\boldsymbol{\mu}}$. The monomial $\binom{\binom{n}{2}}{k} w(R)^{2k} \bar{w}(R)^{2(\binom{n}{2}-k)}$ is equal to $A_{\boldsymbol{\mu}} \mathrm{w}_{\boldsymbol{\mu}}$, where $\boldsymbol{\mu}$ is such that $|R| = 2k$, $\mathrm{w}_{\boldsymbol{\mu}} = w(R)^{2k} \bar{w}(R)^{2(\binom{n}{2}-k)}$ and hence $A_{\boldsymbol{\mu}} = \binom{\binom{n}{2}}{k}$. This procedure of inferring $\boldsymbol{\mu}$ from monomials is trivially implied from Theorem 1. All WFOMC algorithms presented in this paper and in literature [18,10,11] only add simple arithmetic operations over the polynomials obtained in Theorem 1. Hence, inferring $\boldsymbol{\mu}$ from monomials in the WFOMC polynomials is always possible. Keeping only the monomials satisfying the condition $\boldsymbol{\mu} \models \Gamma$ can then be used to impose any cardinality constraint Γ as done in Equation (62). For instance, for a cardinality constraint Γ, one can simply eliminate the terms $\binom{\binom{n}{2}}{k} w(R)^{2k} \bar{w}(R)^{2(\binom{n}{2}-k)} = A_{\boldsymbol{\mu}} \mathrm{w}_{\boldsymbol{\mu}}$ such that $\boldsymbol{\mu} \not\models \Gamma$. Leaving us with $\sum_{\boldsymbol{\mu} \models \Gamma} A_{\boldsymbol{\mu}} \mathrm{w}_{\boldsymbol{\mu}}$.

In general, one can always consider the predicate weights as variables, and the WFOMC algorithms will produce a polynomial on such weights. Once simplified, this polynomial can be expressed as in the RHS of equation (61), with the difference that now the coefficients $A_{\boldsymbol{\mu}}$ are known (since they are produced by the WFOMC algorithms and the polynomial simplification), while the weights $\mathrm{w}_{\boldsymbol{\mu}}$ are symbolic. Thus, in our implementation, instead of computing the coefficients $A_{\boldsymbol{\mu}}$ as in the proof of Theorem 2, we proceed in the following manner: with sympy Python library, we represent symbolically the predicate weights, and then perform the WFOMC operations as operations between polynomials on such variables. This produces a polynomial that when simplified gives us directly the coefficients $A_{\boldsymbol{\mu}}$ and hence a simple way to impose any cardinality constraint as done in Equation (62).

Appendix B. Appendix to Section 5

Lemma 3. *An interpretation ω is a model of $\Psi_{[m]}$ according to Definition 6 if and only if ω satisfies conditions C1-C4 in Lemma 2, where we instantiate $axiom'$, $axiom''$ and $\Theta(x,y)$ as follows:*

- $axiom' := \forall xy.\neg R(x,y)$
- $axiom'' := Acyclic(R)$
- $\Theta(x,y) := \neg R(y,x)$

Proof. If $\omega \models \Psi_{[m]}$ according to Definition 6, since nodes in $[m]$ have zero indegree, we have that $\omega \downarrow [m] \models axiom'$ (C1). Since any subgraph of an acyclic graph is acyclic, we also have that $\omega \downarrow [\bar{m}] \models axiom''$ (C2). And since nodes in $[m]$ have indegree zero, there can be no arrow from $[\bar{m}]$ to $[m]$, i.e., $\omega \models \forall x \in [m]\ \forall y \in [\bar{m}].\Theta(x,y)$ (C3). Finally, C4 is satisfied by construction. On the other hand, if ω satisfies C1 and C3 with the previous instantiations, then the nodes in $[m]$ cannot have incoming arrows. And the graph obtained from a DAG by adding m nodes without incoming arrows is still a DAG. □

Appendix C. Appendix to Section 6

Lemma 4. *An interpretation ω is a model of $\Psi_{[m]}$ according to Definition 9 if and only if ω satisfies conditions C1-C4 in Lemma 2, where we instantiate $axiom'$, $axiom''$ and $\Theta(x,y)$ as follows:*

- $axiom' := Connected(R)$
- $axiom'' := \top$
- $\Theta(x,y) := \neg R(x,y)$

Proof. If $\omega \models \Psi_{[m]}$ according to Definition 9, since $[m]$ is a connected component, we have that $\omega \downarrow [m] \models axiom'$. Also, $axiom''$ is vacuously satisfied. Since $[m]$ is a connected component of ω_R, there can not be an R-edge between $[m]$ and $[\bar{m}]$ – as that would make $[m]$ part of a larger connected subgraph, leading to a contradiction. Hence, $\omega \models \forall x \in [m]\ \forall y \in [\bar{m}].\Theta(x,y)$. Finally, C4 is satisfied by construction. On the other hand, if ω satisfies C4, then it satisfies the condition in Remark 4 for R to be a symmetric and antireflexive relation. So, by satisfying also C1 and C3, the nodes in $[m]$ must be connected and there cannot be edges between them and the other $n-m$ nodes. This means that precisely that $\omega_R \downarrow [m]$ forms a connected component of ω_R. □

C.1. Analysis of the algorithm and proof of Theorem 6

We provide Algorithm 3 that takes as input $\Psi := \forall xy.\Phi(x,y) \wedge Connected(R)$ and $\boldsymbol{k}$ – where $\Phi(x,y)$ is a quantifier-free FO^2 formula and $\boldsymbol{k}$ is a 1-type cardinality vector with $|\boldsymbol{k}| = n$ – and returns $\text{WFOMC}(\Psi, \boldsymbol{k})$. It can be seen that the algorithm runs in polynomial time using much of the same analysis that was used for Algorithm 2. The key ideas being that: the **for** loops in line 5-8 and line 12-14 run both polynomially many iterations w.r.t n; the lexicographical order in the **for** loop on line 5 ensures that the values $A[\boldsymbol{s}']$ required in the function $\overline{\text{WFOMC}}(\Psi_{[m]}, \boldsymbol{s})$ are always already stored in A; and $\text{WFOMC}(\forall xy.\Phi(x,y), \boldsymbol{s}'')$ is an FO^2 WFOMC problem, again computable in polynomial time.

Algorithm 3 WFOMC-Connected.

1: **Input**: $\Psi := \forall xy.\Phi(x,y) \wedge Connected(R), \boldsymbol{k}$
2: **Output**: $\text{WFOMC}(\Psi, \boldsymbol{k})$
3: $A[\boldsymbol{0}] \leftarrow 0$ ▷ A has u indices
4: ▷ $\boldsymbol{0} = \langle 0, ..., 0 \rangle$
5: **for** $\boldsymbol{0} < \boldsymbol{p} \leq \boldsymbol{k}$ where $\boldsymbol{p} \in \mathbb{N}_0^u$ **do** ▷ $\leq$ is Lexical order
6: $\quad A[\boldsymbol{p}] \leftarrow \text{WFOMC}(\Psi'', \boldsymbol{p}) - \frac{1}{|\boldsymbol{p}|} \sum_{m=1}^{|\boldsymbol{p}|-1} \binom{|\boldsymbol{p}|}{m} \cdot m \cdot \overline{\text{WFOMC}}(\Psi_{[m]}, \boldsymbol{p})$
7: **end for**
8: **return** $A[\boldsymbol{k}]$
9:
10: **function** $\overline{\text{WFOMC}}(\Psi_{[m]}, \boldsymbol{s})$ ▷ Equation (34)
11: $\quad S = 0$
12: $\quad$ **for** $\boldsymbol{s}' + \boldsymbol{s}'' = \boldsymbol{s}$ and $|\boldsymbol{s}'| = m$ **do**
13: $\quad\quad S \leftarrow S + A[\boldsymbol{s}'] \cdot \text{WFOMC}(\Psi'', \boldsymbol{s}'') \cdot \prod_{i,j \in [u]} r_{ij}^{s_i' s_j''}$
14: $\quad$ **end for**
15: $\quad$ **return** S
16: **end function**

In line 3, an array A with u indices is initiated and $A[\boldsymbol{0}]$ is assigned the value 0, where $\boldsymbol{0}$ corresponds to the u-dimensional zero vector. The for loop in lines $5-7$ incrementally computes $\text{WFOMC}(\Psi, \boldsymbol{p})$, using equation (36), where the loop runs over all u-dimensional integer vectors $\boldsymbol{p}$, such that $\boldsymbol{p} \leq \boldsymbol{k}$, where $\leq$ is the lexicographical order. The number of possible $\boldsymbol{p}$ vectors is at most n^u. Hence, the for loop runs at most n^u iterations. In line 6, we compute $\text{WFOMC}(\Psi, \boldsymbol{p})$ as given in equation (36). Also in line 6, the function $\overline{\text{WFOMC}}(\Psi_{[m]}, \boldsymbol{p})$ — that computes $\text{WFOMC}(\Psi_{[m]}, \boldsymbol{p})$ —is called at most $|\boldsymbol{p}| - 1$ times, which is bounded above by n. $A[\boldsymbol{p}]$ stores the value $\text{WFOMC}(\Psi, \boldsymbol{p})$. In the function $\overline{\text{WFOMC}}(\Psi_{[m]}, \boldsymbol{s})$, the number of iterations in the for loop is bounded above by n^{2u}. And $\text{WFOMC}(\Phi, \boldsymbol{s}'')$ is an FO^2 WFOMC problem, again computable in polynomial time. Hence, the algorithm WFOMC-Connected runs in polynomial time w.r.t domain cardinality. Notice that since loop 5-7 runs in lexicographical order, the $A[\boldsymbol{s}']$ required in the function $\overline{\text{WFOMC}}(\Psi_{[m]}, \boldsymbol{s})$ are always already stored in A.

Summing $\text{WFOMC}(\Psi, \boldsymbol{k})$ over all possible $\boldsymbol{k}$ such that $|\boldsymbol{k}| = n$, we can compute $\text{WFOMC}(\Psi, n)$ in polynomial time w.r.t domain cardinality n. Moreover, due to the modularity of the skolemization process for WFOMC [5], we can extend this result to prove the domain liftability of the entire FO^2 fragment, with connectivity axiom. Using Theorem 2 and Remark 2, we can also extend domain-liftability of FO^2, with connectivity axiom and cardinality constraints. Finally, since WFOMC of any C^2 formula can be modularly reduced to WFOMC of an FO^2 formula with cardinality constraints [18], we have Theorem 6.

Appendix D. Appendix to Section 7

Lemma 5. *An interpretation ω is a model of $\Psi_{[m]}$ according to Definition 14 if and only if ω satisfies conditions C1-C4 in Lemma 2, where we instantiate $axiom'$, $axiom''$ and $\Theta(x,y)$ as follows:*

- $axiom' := Tree(R)$
- $axiom'' := Forest(R)$
- $\Theta(x,y) := \neg R(x,y)$

Proof. If $\omega \vDash \Psi_{[m]}$ according to Definition 14, since $\omega_R \downarrow [m]$ is a tree, we have that $\omega \downarrow [m] \vDash axiom'$. Also, any subgraph of a forest is still a forest, so $\omega \downarrow [\bar{m}] \vDash axiom''$. Since $[m]$ is a connected component of ω_R, there can not be an R-edge between $[m]$ and $[\bar{m}]$. Hence, $\omega \vDash \forall x \in [m]\ \forall y \in [\bar{m}].\Theta(x,y)$. Finally, C4 is satisfied by construction. On the other hand, if ω satisfies C4, then it satisfies the condition in Remark 8 for R to be a symmetric and antireflexive relation. Hence, by satisfying also C1, C2 and C3, the nodes in $[m]$ form a tree (C1), those in $[\bar{m}]$ form a forest (C2), and the two subgraphs are disconnected (C3). This implies that $\omega \vDash \Psi_{[m]}$ according to Definition 14. □

Algorithm 4 WFOMC-Forest.

Input: $\Psi := \forall xy.\Phi(x,y) \wedge Forest(R), \boldsymbol{k}$
Output: $\text{WFOMC}(\Psi, \boldsymbol{k})$
$A[\mathbf{0}] \leftarrow 1$ ▷ A has u indices
▷ $\mathbf{0} = \langle 0, ..., 0 \rangle$
for $\mathbf{0} < \boldsymbol{p} \leq \boldsymbol{k}$ where $\boldsymbol{p} \in \mathbb{N}_0^u$ **do** ▷ Lexical order
  $A[\boldsymbol{p}] \leftarrow \sum_{m=1}^{|\boldsymbol{p}|} \binom{|\boldsymbol{p}|-1}{m-1} \overline{\text{WFOMC}}(\Psi_{[m]}, \boldsymbol{p})$
end for
return $A[\boldsymbol{k}]$

function $\overline{\text{WFOMC}}(\Psi_{[m]}, \boldsymbol{s})$ ▷ Equation (46)
  $S = 0$
  for $\boldsymbol{s}' + \boldsymbol{s}'' = \boldsymbol{s}$ and $|\boldsymbol{s}'| = m$ **do**
    $S \leftarrow S + \text{WFOMC}(\Psi', \boldsymbol{s}') \cdot A[\boldsymbol{s}''] \cdot \prod_{i,j \in [u]} r_{ij}^{s'_i s''_j}$
  end for
  return S
end function

Appendix E. Appendix to Section 8

Table 2
Number of active 1-types (# 1-types) and presence of cardinality constrains on binary predicates (cardinality $R/2$) for each combinatorial experiment, after modular reduction to FO^2 with cardinality constraints and WFOMC-preserving skolemization. A clear negative correlation between them and the maximum domain size having FOMC runtime within 100 seconds (max n) can be observed.

Combinatorial problem	# 1-types	cardinality $R/2$	max n
DAGs w/ $2n$ edges (23)	1	yes	65
DAGs w/ $2n$ edges and one source (25)	3	yes	29
DAGs w/ one source (26)	3	no	122
DAGs w/ one source and one sink (27)	9	no	20
trees (39)	1	yes	79
3-colored connected graphs (42)	3	no	60
forests w/o isolated nodes (50)	2	yes	36

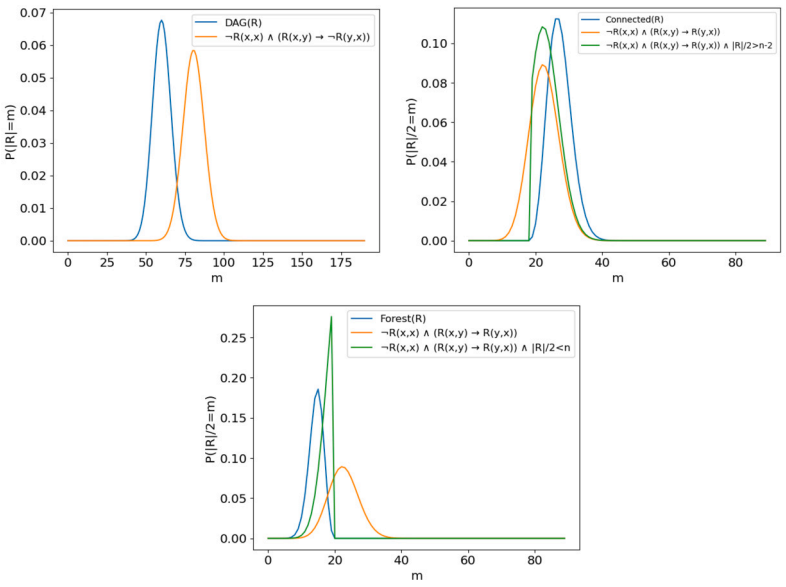


Fig. 5. Probability distributions of the number of edges of the directed/undirected graphs produced by MLNs with only the predicate $R/2$. The legends report the hard constraints, while the soft constraint is always $-1 : R(x,y)$. The domain size is fixed to $n = 20$.

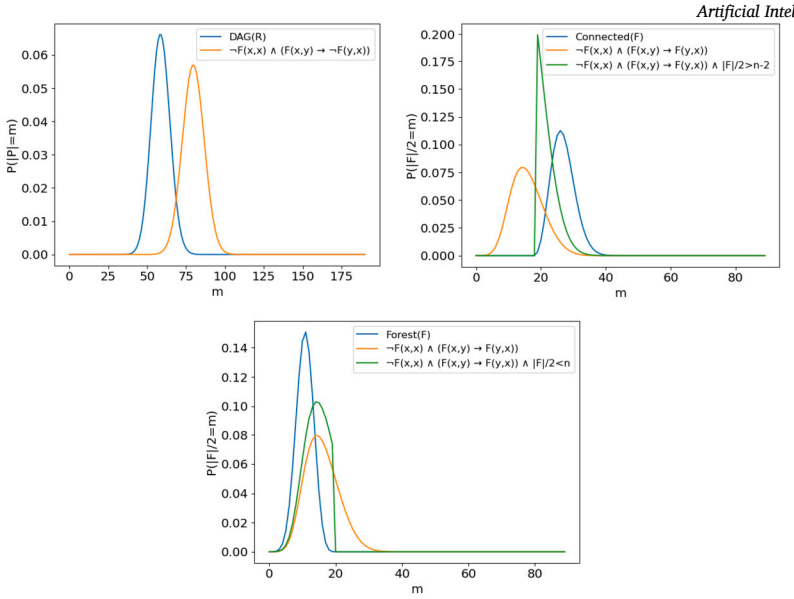


Fig. 6. Probability distributions of the number of edges of the graphs produced by predicate F with "smokers and friends" MLNs. The legends report the hard constraints, while the soft constraints are always $0 : S(x)$, $-1 : F(x, y)$ and $3 : P(x, y)$. The domain size is fixed to $n = 20$.

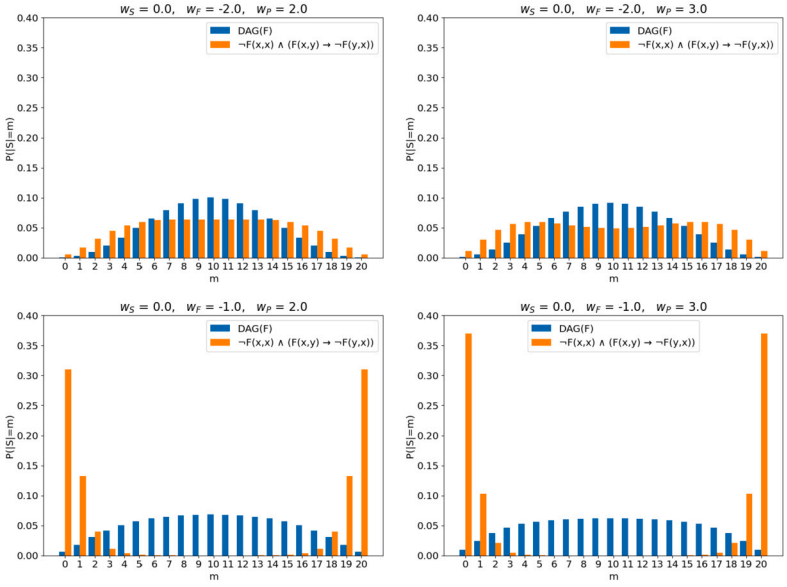


Fig. 7. Distributions of smokers for different weight values. The legends report the hard constraints, while the titles report the weights of the soft constraints. The domain size is fixed to $n = 20$.

Data availability

The code for the paper is online.

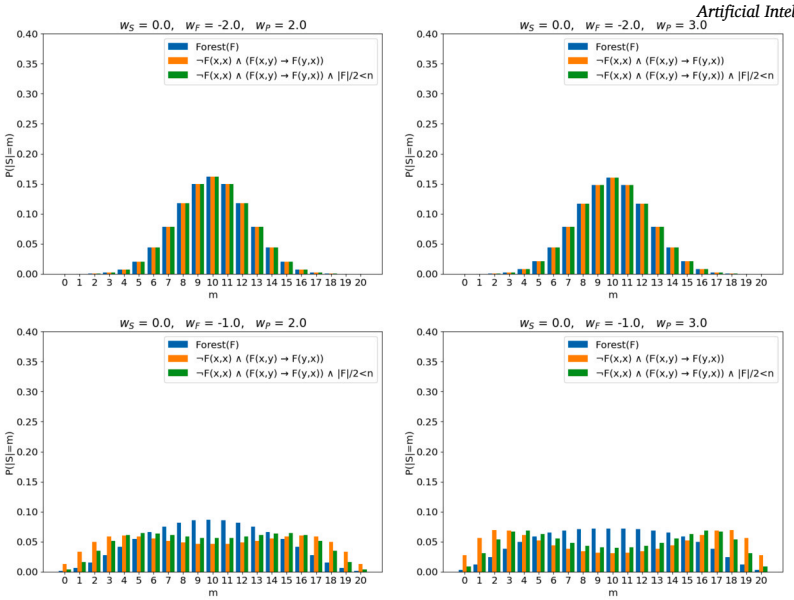


Fig. 8. Distributions of smokers for different weight values. The legends report the hard constraints, while the titles report the weights of the soft constraints. The domain size is fixed to $n = 20$.

References

[1] L. Getoor, B. Taskar, Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning), The MIT Press, 2007.
[2] L.D. Raedt, K. Kersting, S. Natarajan, D. Poole, Statistical Relational Artificial Intelligence: Logic, Probability, and Computation, Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan & Claypool Publishers, 2016.
[3] M. Richardson, P. Domingos, Markov logic networks, Mach. Learn. 62 (1–2) (2006) 107–136.
[4] D. firens, G. Van Den Broeck, J. Renkens, D. Shterionov, B. Gutmann, I. Thon, G. Janssens, L. De Raedt, Inference and learning in probabilistic logic programs using weighted Boolean formulas, Theory Pract. Log. Program. 15 (3) (2015) 358–401, https://doi.org/10.1017/S1471068414000076.
[5] G.V. den Broeck, W. Meert, A. Darwiche, Skolemization for weighted first-order model counting, in: C. Baral, G.D. Giacomo, T. Eiter (Eds.), Principles of Knowledge Representation and Reasoning: Proceedings of the Fourteenth International Conference, KR 2014, Vienna, Austria, July 20-24, 2014, AAAI Press, 2014, http://www.aaai.org/ocs/index.php/KR/KR14/paper/view/8012.
[6] V. Gogate, P.M. Domingos, Probabilistic theorem proving, in: F.G. Cozman, A. Pfeffer (Eds.), UAI 2011, Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence, Barcelona, Spain, July 14–17, 2011, AUAI Press, 2011, pp. 256–265.
[7] G.V. den Broeck, On the completeness of first-order knowledge compilation for lifted probabilistic inference, in: J. Shawe-Taylor, R.S. Zemel, P.L. Bartlett, F.C.N. Pereira, K.Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a Meeting Held 12-14 December 2011, Granada, Spain, vol. 24, Curran Associates, Inc., 2011, pp. 1386–1394, https://proceedings.neurips.cc/paper/2011/hash/846c260d715e5b854ffad5f70a516c88-Abstract.html.
[8] P. Beame, G.V. den Broeck, E. Gribkoff, D. Suciu, Symmetric weighted first-order model counting, in: T. Milo, D. Calvanese (Eds.), Proceedings of the 34th ACM Symposium on Principles of Database Systems, PODS 2015, Melbourne, Victoria, Australia, May 31 - June 4, 2015, ACM, 2015, pp. 313–328.
[9] A. Kuusisto, C. Lutz, Weighted model counting beyond two-variable logic, in: A. Dawar, E. Grädel (Eds.), Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2018, Oxford, UK, July 09-12, 2018, ACM, 2018, pp. 619–628.
[10] S. Malhotra, L. Serafini, A combinatorial approach to weighted model counting in the two-variable fragment with cardinality constraints, in: AIxIA 2021 – Advances in Artificial Intelligence: 20th International Conference of the Italian Association for Artificial Intelligence, Virtual Event, December 1–3, 2021, Revised Selected Papers, Springer-Verlag, Berlin, Heidelberg, 2021, pp. 137–152.
[11] S. Malhotra, L. Serafini, Weighted model counting in fo2 with cardinality constraints and counting quantifiers: a closed form formula, Proc. AAAI Conf. Artif. Intell. 36 (5) (2022) 5817–5824, https://doi.org/10.1609/aaai.v36i5.20525, https://ojs.aaai.org/index.php/AAAI/article/view/20525.
[12] J. Barvínek, T. van Bremen, Y. Wang, F. Železný, O. Kuželka, Automatic conjecturing of p-recursions using lifted inference, in: N. Katzouris, A. Artikis (Eds.), Inductive Logic Programming, Springer International Publishing, Cham, 2022, pp. 17–25.
[13] M. Svatoš, P. Jung, J. Tóth, Y. Wang, O. Kuželka, On discovering interesting combinatorial integer sequences, in: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI '23, 2023.
[14] G.V. den Broeck, N. Taghipour, W. Meert, J. Davis, L.D. Raedt, Lifted probabilistic inference by first-order knowledge compilation, in: T. Walsh (Ed.), IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011, AAAI Press/International Joint Conferences on Artificial Intelligence, IJCAI/AAAI, 2011, pp. 2178–2185.
[15] S.M. Kazemi, A. Kimmig, G.V. den Broeck, D. Poole, Domain recursion for lifted inference with existential quantifiers, CoRR, arXiv:1707.07763, 2017, pp. 1386–1394, http://arxiv.org/abs/1707.07763.
[16] S.M. Kazemi, A. Kimmig, G.V. den Broeck, D. Poole, New liftable classes for first-order probabilistic inference, in: D.D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, R. Garnett (Eds.), Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain, 2016, pp. 3117–3125, https://proceedings.neurips.cc/paper/2016/hash/c88d8d0a6097754525e02c2246d8d27f-Abstract.html.
[17] M. Jaeger, G. Van den Broeck, Liftability of probabilistic inference: Upper and lower bounds, 2012-08-18.
[18] O. Kuzelka, Weighted first-order model counting in the two-variable fragment with counting quantifiers, J. Artif. Intell. Res. 70 (2021) 1281–1307, https://doi.org/10.1613/jair.1.12320.
[19] J. Tóth, O. Kuzelka, Lifted inference with linear order axiom, in: B. Williams, Y. Chen, J. Neville (Eds.), Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023, AAAI Press, 2023, pp. 12295–12304.

[20] T. van Bremen, O. Kuželka, Lifted inference with tree axioms, in: Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning, 2021, pp. 599–608.
[21] J.R. Clough, J. Gollings, T.V. Loach, T.S. Evans, Transitive reduction of citation networks, J. Complex Netw. 3 (2) (2014) 189–203, https://doi.org/10.1093/comnet/cnu039, https://academic.oup.com/comnet/article-pdf/3/2/189/1071092/cnu039.pdf.
[22] J. Ugander, B. Karrer, L. Backstrom, C.A. Marlow, The anatomy of the Facebook social graph, arXiv:1111.4503, 2011.
[23] I.M. Gessel, Counting acyclic digraphs by sources and sinks, Discrete Math. 160 (1996) 253–258.
[24] R.W. Robinson, Counting labeled acyclic digraphs, in: New Directions in the Theory of Graphs, 1973, pp. 239–273.
[25] L. Takács, On Cayley's formula for counting forests, J. Comb. Theory, Ser. A 53 (2) (1990) 321–323, https://doi.org/10.1016/0097-3165(90)90064-4.
[26] N. Immerman, Descriptive Complexity, Springer Science & Business Media, 2012.
[27] H.S. Wilf, Generatingfunctionology, A. K. Peters, Ltd., USA, 2006.
[28] F. Harary, E.M. Palmer, Graphical Enumeration, Addison-Wesley, 1973.
[29] F. Bienvenu, A. Lambert, M. Steel, Combinatorial and stochastic properties of ranked tree-child networks, Random Struct. Algorithms 60 (4) (2022) 653–689, https://doi.org/10.1002/rsa.21048, arXiv:2007.09701 [math, q-bio], http://arxiv.org/abs/2007.09701.
[30] G. Cardona, L. Zhang, Counting and enumerating tree-child networks and their subclasses, J. Comput. Syst. Sci. 114 (2020) 84–104, https://doi.org/10.1016/j.jcss.2020.06.001, https://linkinghub.elsevier.com/retrieve/pii/S0022000020300611.
[31] M. Fuchs, H. Liu, G.-R. Yu, A short note on the exact counting of tree-child networks, arXiv:2110.03842 [math, q-bio], Oct. 2021, http://arxiv.org/abs/2110.03842.
[32] T. Hinrichs, M. Genesereth, Herbrand logic, LG-2006-02, Stanford Reports, 2009, https://www.cs.uic.edu/~hinrichs/papers/hinrichs2006herbrand.pdf.
[33] E. Gradel, M. Otto, E. Rosen, Two-variable logic with counting is decidable, in: Proceedings of Twelfth Annual IEEE Symposium on Logic in Computer Science, IEEE, 1997, pp. 306–317.
[34] S. Malhotra, L. Serafini, On projectivity in Markov logic networks, in: Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2022, Grenoble, France, September 19–23, 2022, Proceedings, Part V, Springer-Verlag, Berlin, Heidelberg, 2023, pp. 223–238.
[35] V. Rodionov, On the number of labeled acyclic digraphs, Discrete Math. 105 (1) (1992) 319–321, https://doi.org/10.1016/0012-365X(92)90155-9, https://www.sciencedirect.com/science/article/pii/0012365X92901559.
[36] OEIS Foundation Inc., The on-line encyclopedia of integer sequences, published electronically at http://oeis.org, 2023.
[37] M. https://mathoverflow.net/users/282217/marcel, Is there a formula for the number of st-dags (dag with 1 source and 1 sink) with n vertices? MathOverflow, https://mathoverflow.net/q/395095, version: 2021-06-11.
[38] M. Mansouri, Counting general phylogenetic networks, Australas. J. Comb. (2022).
[39] M. Fuchs, B. Gittenberger, M. Mansouri, Counting phylogenetic networks with few reticulation vertices: exact enumeration and corrections, arXiv:2006.15784 [math], Mar. 2021, http://arxiv.org/abs/2006.15784.
[40] M. Pons, J. Batle, Combinatorial characterization of a certain class of words and a conjectured connection with general subclasses of phylogenetic tree-child networks, Sci. Rep. 11 (1) (2021) 1–14.
[41] C. McDiarmid, C. Semple, D. Welsh, Counting phylogenetic networks, Ann. Comb. 19 (1) (2015) 205–224, https://doi.org/10.1007/s00026-015-0260-2, http://link.springer.com/10.1007/s00026-015-0260-2.
[42] D.H. Huson, R. Rupp, C. Scornavacca, Phylogenetic Networks: Concepts, Algorithms and Applications, Cambridge University Press, 2010.
[43] M. Fuchs, E.-Y. Huang, G.-R. Yu, Counting phylogenetic networks with few reticulation vertices: a second approach, Discrete Appl. Math. 320 (2022) 140–149.
[44] Y.-S. Chang, M. Fuchs, H. Liu, M. Wallner, G.-R. Yu, Enumeration of d-combining tree-child networks, in: M.D. Ward (Ed.), 33rd International Conference on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms (AofA 2022), in: Leibniz International Proceedings in Informatics (LIPIcs), vol. 225, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 2022, pp. 5:1–5:13, https://drops.dagstuhl.de/opus/volltexte/2022/16091.
[45] M. Fuchs, G.-R. Yu, L. Zhang, On the asymptotic growth of the number of tree-child networks, Eur. J. Comb. 93 (2021) 103278.
[46] M. Bouvel, P. Gambette, M. Mansouri, Counting phylogenetic networks of level 1 and 2, J. Math. Biol. 81 (6–7) (2020) 1357–1395.
[47] M. Bona, Handbook of Enumerative Combinatorics, Discrete Mathematics and Its Applications, CRC Press, 2015, https://books.google.it/books?id=j3kZBwAAQBAJ.
[48] R.C. Read, E.M. Wright, Coloured graphs: a correction and extension, Can. J. Math. 22 (3) (1970) 594–596, https://doi.org/10.4153/CJM-1970-066-1.
[49] J.-C. Fournier, Graph Theory and Applications: With Exercises and Problems, John Wiley & Sons, 2013.
[50] J. Tóth, O. Kuželka, Complexity of weighted first-order model counting in the two-variable fragment with counting quantifiers: a bound to beat, in: Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning, 2024, pp. 676–686.
[51] T. van Bremen, O. Kuželka, Faster lifting for two-variable logic using cell graphs, in: C. de Campos, M.H. Maathuis (Eds.), Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence, in: Proceedings of Machine Learning Research, vol. 161, PMLR, 2021, pp. 1393–1402, https://proceedings.mlr.press/v161/bremen21a.html.