New Liftable Classes for First-Order Probabilistic Inference

Authored by:

David Poole
Guy Van den Broeck
Seyed Mehran Kazemi
Angelika Kimmig

Abstract

Statistical relational models provide compact encodings of probabilistic dependencies in relational domains, but result in highly intractable graphical models. The goal of lifted inference is to carry out probabilistic inference without needing to reason about each individual separately, by instead treating exchangeable, undistinguished objects as a whole. In this paper, we study the domain recursion inference rule, which, despite its central role in early theoretical results on domain-lifted inference, has later been believed redundant. We show that this rule is more powerful than expected, and in fact significantly extends the range of models for which lifted inference runs in time polynomial in the number of individuals in the domain. This includes an open problem called S4, the symmetric transitivity model, and a first-order logic encoding of the birthday paradox. We further identify new classes S2FO2 and S2RU of domain-liftable theories, which respectively subsume FO2 and recursively unary theories, the largest classes of domain-liftable theories known so far, and show that using domain recursion can achieve exponential speedup even in theories that cannot fully be lifted with the existing set of inference rules.

1 Paper Body

Statistical relational learning (SRL) [8] aims at unifying logic and probability for reasoning and learning in noisy domains, described in terms of individuals (or objects), and the relationships between them. Statistical relational models [10], or template-based models [18] extend Bayesian and Markov networks with individuals and relations, and compactly describe probabilistic dependencies among them. These models encode exchangeability among the objects: individuals that we have the same information about are treated similarly. A key challenge with SRL models is the fact that they represent highly intractable, densely connected graphical models, typically with millions of random variables. The aim of lifted

inference [23] is to carry out probabilistic inference without needing to reason about each individual separately, by instead treating exchangeable, undistinguished objects as a whole. Over the past decade, a large number of lifted inference rules have been proposed [5, 9, 11, 14, 20, 22, 28, 30], often providing exponential speedups for specific SRL models. These basic exact inference techniques have applications in (tractable) lifted learning [32], where the main task is to efficiently compute partition functions, and in variational and over-symmetric approximations [29, 33]. Moreover, they provided the foundation for a rich literature on approximate lifted inference and learning [1, 4, 13, 17, 19, 21, 25, 34]. 30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

The theoretical study of lifted inference began with the complexity notion of domain-lifted inference [31] (a concept similar to data complexity in databases). Inference is domain-lifted when it runs in time polynomial in the number of individuals in the domain. By identifying liftable classes of models, guaranteeing domain-lifted inference, one can characterize the theoretical power of the various inference rules. For example, the class FO 2 , encoding dependencies among pairs of individuals (i.e., two logical variables), is liftable [30]. Kazemi and Poole [15] introduce a liftable class called recursively unary, capturing hierarchical simplification rules. Beame et al. [3] identify liftable classes of probabilistic database queries. Such results elevate the specific inference rules and examples to a general principle, and bring lifted inference in line with complexity and database theory [3]. This paper studies the domain recursion inference rule, which applies the principle of induction on the domain size. The rule makes one individual A in the domain explicit. Afterwards, the other inference rules simplify the SRL model up to the point where it becomes identical to the original model, except the domain size has decreased. Domain recursion was introduced by Van den Broeck [31] and was central to the proof that FO 2 is liftable. However, later work showed that simpler rules suffice to capture FO 2 [27], and the domain recursion rule was forgotten. We show that domain recursion is more powerful than expected, and can lift models that are otherwise not amenable to domain-lifted inference. This includes an open problem by Beame et al. [3], asking for an inference rule for a logical sentence called S4. It also includes the symmetric transitivity model, and an encoding of the birthday paradox in first-order logic. There previously did not exist any efficient algorithm to compute the partition function of these SRL models, and we obtain exponential speedups. Next, we prove that domain recursion supports its own large classes of liftable models S 2 FO 2 subsuming FO 2 , and S 2 RU subsuming recursive unary1 . All existing exact lifted inference algorithms (e.g., [11, 15, 28]) resort to grounding the theories in S 2 FO 2 or S 2 RU that are not in FO 2 or recursively unary, and require time exponential in the domain size. These results will be established using the weighted first-order model counting (WFOMC) formulation of SRL models [28]. WFOMC is close to classical first-order logic, and it can encode many other SRL models, including Markov logic [24], parfactor graphs [23], some probabilistic programs [7], relational Bayesian networks [12], and probabilistic databases [26]. It is a basic specification language that

simplifies the development of lifted inference algorithms [3, 11, 28].

2

Background and Notation

A population is a set of constants denoting individuals (or objects). A logical variable (LV) is typed with a population. We represent LVs with lower-case letters, constants with upper-case letters, the population associated with a LV x with ?x , and its cardinality with —?x —. That is, a population ?x is a set of constants {X1 , . . . , Xn }, and we use x ? ?x as a shorthand for instantiating x with one of the Xi . A parametrized random variable (PRV) is of the form F(t1 , . . . , tk) where F is a predicate symbol and each ti is a LV or a constant. A unary PRV contains exactly one LV and a binary PRV contains exactly two LVs. A grounding of a PRV is obtained by replacing each of its LVs x by one of the individuals in ?x . A literal is a PRV or its negation. A formula ? is a literal, a disjunction ?1 ? ?2 of formulas, a conjunction ?1 ? ?2 of formulas, or a quantified formula ?x ? ?x : ?(x) or ?x ? ?x : ?(x) where x appears in ?(x). A sentence is a formula with all LVs quantified. A clause is a disjunction of literals. A theory is a set of sentences. A theory is clausal if all its sentences are clauses. An interpretation is an assignment of values to all ground PRVs in a theory. An interpretation I is a model of a theory T , I —= T , if given its value assignments, all sentences in T evaluate to True. Let F(T) be the set of predicate symbols in theory T , and ? : F(T) ? R and ? : F(T) ? R be two functions that map each predicate F to weights. These functions associate a weight with assigning True or False to ground PRVs F(C1 , . . . , Ck). For an interpretation I of T , let ? T rue F alse be the set of ground the ones assigned False. The weight of I is Q PRVs assigned True, and Q? given by ?(I) = F(C1 ,...,Ck)??T rue ?(F) ? F(C1 ,...,Ck)??F alse ?(F). Given a theory T and two functions ? and ?, the weighted first-order model count (WFOMC) of the theory given ? and ? P is: WFOMC(T —?, ?) = I—=T ?(I). 1

All proofs can be found in the extended version of the paper at: https://arxiv.org/abs/1610.08445

2

In this paper, we assume that all theories are clausal and do not contain existential quantifiers. The latter can be achieved using the Skolemization procedure of Van den Broeck et al. [30], which efficiently transforms a theory T with existential quantifiers into a theory T 0 without existential quantifiers that has the same weighted model count. That is, our theories are sets of finite-domain, function-free first-order clauses whose LVs are all universally quantified (and typed with a population). Furthermore, when a clause mentions two LVs x1 and x2 with the same population ?x , or a LV x with population ?x and a constant C ? ?x , we assume they refer to different individuals.2 Example 1. Consider the theory ?x ? ?x : ?Smokes(x) ? Cancer(x) having only one clause and assume ?x = {A, B}. The assignment Smokes(A) = True, Smokes(B) = False, Cancer(A) = True, Cancer(B) = True is a model. Assuming ?(Smokes) = 0.2, ?(Cancer) = 0.8, ?(Smokes) = 0.5 and ?(Cancer) = 1.2, the weight of this model is 0.2 ? 0.5 ? 0.8 ? 0.8. This theory has eight other models. The WFOMC can be calculated by summing the weights of all nine models. 2.1

Converting Inference for SRL Models into WFOMC

For many SRL models, (lifted) inference can be converted into a WFOMC problem. As an example, consider a Markov logic network (MLN) [24] with weighted formulae (w1 : F1 , . . . , wk : Fk). For every weighted formula wi : Fi of this MLN, let theory T have a sentence Auxi (x, . . .) ? Fi such that Auxi is a predicate having all LVs appearing in Fi . Assuming ?(Auxi) = exp(wi), and ? and ? are 1 for the other predicates, the partition function of the MLN is equal to WFOMC(T). 2.2

Calculating the WFOMC of a Theory

We now describe a set of rules R that can be applied to a theory to find its WFOMC efficiently; for more details, readers are directed to [28], [22] or [11]. We use the following theory T with two clauses and four PRVs (S(x, m), R(x, m), T(x) and Q(x)) as our running example: ?x ? ?x , m ? ?m : Q(x) ? R(x, m) ? S(x, m)

?x ? ?x , m ? ?m : S(x, m) ? T(x)

Lifted Decomposition Assume we ground x in T . Then the clauses mentioning an arbitrary Xi ? ?x are ?m ? ?m : Q(Xi) ? R(Xi , m) ? S(Xi , m) and ?m ? ?m : S(Xi , m) ? T(Xi). These clauses are totally disconnected from clauses mentioning Xj ? ?x (j 6= i), and are the same up to renaming Xi to Xj . Given the exchangeability of the individuals, we can calculate the WFOMC of only the clauses mentioning Xi and raise the result to the power of the number of connected components (—?x —). Assuming T1 is the theory that results from substituting x with Xi , WFOMC(T) = WFOMC(T1)—?x — . Case-Analysis The WFOMC of T1 can be computed by a case analysis over different assignments of values to a ground PRV, e.g., Q(Xi). Let T2 and T3 represent T1 ? Q(Xi) and T1 ? ?Q(Xi) respectively. Then, WFOMC(T1) = WFOMC(T2) + WFOMC(T3). We follow the process for T3 (the process for T2 will be similar) having clauses ?Q(Xi), ?m ? ?m : Q(Xi) ? R(Xi , m) ? S(Xi , m) and ?m ? ?m : S(Xi , m) ? T(Xi). Unit Propagation When a clause in the theory has only one literal, we can propagate the effect of this clause through the theory and remove it3 . In T3 , ?Q(Xi) is a unit clause. Having this unit clause, we can simplify the second clause and get the theory T4 having clauses ?m ? ?m : R(Xi , m) ? S(Xi , m) and ?m ? ?m : S(Xi , m) ? T(Xi). Lifted Case-Analysis Case-analysis can be done for PRVs having one logical variable in a lifted way. Consider the S(Xi , m) in T4 . Due to the exchangeability of the individuals, we do not have to consider all possible assignments to all ground PRVs of S(Xi , m), but only the ones where the number of individuals M ? ?m for which S(Xi , M) is True (or equivalently False) is different. This means considering —?m — + 1 cases suffices, corresponding to S(Xi , M) being True for exactly

j = 0, . . . , —?m — individuals. Note that we must multiply by —?jm — to account for the number 2

Equivalently, we can disjoin x1 = x2 or x = C to the clause. Note that unit propagation may remove clauses and random variables from the theory. To account for them, smoothing multiplies the WFOMC by 2#rv , where #rv represents the number of removed variables. 3

3

of ways one can select j out of —?m — individuals. Let T4j represent T4 with two more clauses: ?m ? ?mT : S(Xi , m) and ?m ? ?mF : ?S(Xi , m), where ?mT represents the j individuals in ?m for which S(Xi , M) is True, and ?mF represents the other —?m — ? j individuals. Then

P—? — WFOMC(T4) = j=0m —?jm — WFOMC(T4j). Shattering In T4j , the individuals in ?m are no longer exchangeable: we know different things about those in ?mT and those in ?mF . We need to shatter every clause having individuals coming from ?m to make the theory exchangeable. To do so, the clause ?m ? ?m : R(Xi , m) ? S(Xi , m) in T4j must be shattered to ?m ? ?mT : R(Xi , m)?S(Xi , m) and ?m ? ?mF : R(Xi , m)?S(Xi , m) (and similarly for the other formulae). The shattered theory T5j after unit propagation will have clauses ?m ? ?mF : R(Xi , m) and ?m ? ?mF : T(Xi). Decomposition, Caching, and Grounding In T5j , the two clauses have different PRVs, i.e., they are disconnected. In such cases, we apply decomposition, i.e., find the WFOMC of each connected component separately and return the product. The WFOMC of the theory can be found by continuing to apply the above rules. In all the above steps, after finding the WFOMC of each (sub-)theory, we store the results in a cache so we can reuse them if the same WFOMC is required again. By following these steps, one can find the WFOMC of many theories in polynomial time. However, if we reach a point where none of the above rules are applicable, we ground one of the populations which makes the process exponential in the number of individuals. 2.3

Domain-Liftability

The following notions allow us to study the power of a set of lifted inference rules. Definition 1. A theory is domain-liftable [31] if calculating its WFOMC is polynomial in —?x1 —, —?x2 —, . . . , —?xk — where x1 , x2 , . . . , xk represent the LVs in the theory. A class C of theories is domain-liftable if ?T ? C, T is domain-liftable. So far, two main classes of domain-liftable theories have been recognized: FO 2 [30, 31] and recursively unary [15, 22]. Definition 2. A theory is in FO 2 if all its clauses have up to two LVs. Definition 3. A theory T is recursively unary (RU) if for every theory T 0 resulting from applying rules in R except for lifted case analysis to T , until no more rules apply, there exists some unary PRV in T 0 and a generic case of lifted case-analysis on this unary PRV is itself RU. Note that the time needed to check whether a theory is in FO 2 or RU is independent of the domain sizes in the theory. For FO 2 , the membership check can be done in time linear in the size of the theory, whereas for RU, only a worst-case exponential procedure is known. Thus, FO 2 currently offers a faster membership check than RU, but as we show later, RU subsumes FO 2 . This gives rise to a trade-off between fast membership checking and modeling power for, e.g., lifted learning purposes.

3

The Domain Recursion Rule

Van den Broeck [31] considered another rule called domain recursion in the set of rules for calculating the WFOMC of a theory. The intuition behind domain recursion is that it modifies a domain ?x by making one element explicit: ?x = ?x0 ? {A} with A 6? ?x0 . Next, clauses are rewritten in terms of ?x0 and

A while removing ?x from the theory entirely. Then, by applying standard rules in R on this modified theory, the problem is reduced to a WFOMC problem on a theory identical to the original one, except that ?x is replaced by the smaller domain ?x0 . This lets us compute WFOMC using dynamic programming. We refer to R extended with the domain recursion rule as RD . Example 2. Suppose we have a theory whose only clause is ?x, y ? ?p : ?Friend(x, y) ? Friend(y, x), stating if x is friends with y, y is also friends with x. One way to calculate the WFOMC of this theory is by grounding only one individual in ?p and then using R. Let A be an individual in ?p and let ?p0 = ?p ? {A}. We can (using domain recursion) rewrite the theory as: ?x ? ?p0 : ?Friend(x, A) ? Friend(A, x), ?y ? ?p0 : ?Friend(A, y) ? Friend(y, A), and ?x, y ? ?p0 : ?Friend(x, y) ? Friend(y, x). Lifted case-analysis on Friend(p0 , A) and Friend(A, p0), 4

shattering and unit propagation give ?x, y ? ?p0 : ?Friend(x, y) ? Friend(y, x). This theory is equivalent to our initial theory, with the only difference being that the population of people has decreased by one. By keeping a cache of the values of each sub-theory, one can verify that this process finds the WFOMC of the above theory in polynomial time. Note that the theory in Example 2 is in FO 2 and as proved in [27], its WFOMC can be computed without using the domain recursion rule4 . This proof has caused the domain recursion rule to be forgotten in the lifted inference community. In the next section, we revive this rule and identify a class of theories that are only domain-liftable when using the domain recursion rule.

4

Domain Recursion Makes More Theories Domain-Liftable

In this section, we show three example theories that are not domain-liftable when using R, yet become domain-liftable with domain recursion. S4 Clause: Beame et al. [3] identified a clause (S4) with four binary PRVs having the same predicate and proved that, even though the rules R in Section 2.2 cannot calculate the WFOMC of that clause, there is a polynomial-time algorithm for finding its WFOMC. They concluded that this set of rules R for finding the WFOMC of theories does not suffice, asking for new rules to compute their theory. We prove that adding domain recursion to the set achieves this goal. Proposition 1. The theory consisting of the S4 clause ?x1 , x2 ? ?x , y1 , y2 ? ?y : S(x1 , y1) ? ?S(x2 , y1) ? S(x2 , y2) ? ?S(x1 , y2) is domain-liftable using RD . Symmetric Transitivity: Domain-liftable calculation of WFOMC for the transitivity formula is a long-standing open problem. Symmetric transitivity is easier as its model count corresponds to the Bell number, but solving it using general-purpose rules has been an open problem. Consider clauses ?x, y, z ? ?p : ?F(x, y) ? ?F(y, z) ? F(x, z) and ?x, y ? ?p : ?F(x, y) ? F(y, x) defining a symmetric transitivity relation. For example, ?p may indicate the population of people and F may indicate friendship. Proposition 2. The symmetric-transitivity theory is domain-liftable using RD . Birthday Paradox: The birthday paradox problem [2] is to compute the probability that in a set of n randomly chosen people, two of them have the same birthday. A first-order encoding of this problem requires computing the WFOMC for a theory with clauses ?p ? ?p , ?d ? ?d : Born(p, d), ?p ? ?p , d1 , d2 ? ?d : ?Born(p, d1

) ? ?Born(p, d2), and ?p1 , p2 ? ?p , d ? ?d : ?Born(p1 , d) ? ?Born(p2 , d), where ?p and ?d represent the population of people and days. The first two clauses impose the condition that every person is born in exactly one day, and the third clause states the ?no two people are born on the same day? query. Proposition 3. The birthday-paradox theory is domain-liftable using RD .

New Domain-Liftable Classes: S 2 FO 2 and S 2 RU

5

In this section, we identify new domain-liftable classes, enabled by the domain recursion rule. Definition 4. Let ?(S) be a clausal theory that uses a single binary predicate S, such that each clause has exactly two different literals of S. Let ? = ?(S1) ? ?(S2) ? ? ? ? ? ?(Sn) where the Si are different binary predicates. Let ? be a theory where all clauses contain at most one Si literal, and the clauses that contain an Si literal contain no other literals with more than one LV. Then, S 2 FO 2 and S 2 RU are the classes of theories of the form ? ? ? where ? ? FO 2 and ? ? RU respectively. Theorem 1. S 2 FO 2 and S 2 RU are domain-liftable using RD . Proof. The case where ? = ? is trivial. Let ? = ?(S1) ? ?(S2) ? ? ? ? ? ? ?(Sn). Once we remove all PRVs having none or one LV by (lifted) case-analysis, the remaining clauses can be divided into n + 1 components: the i-th component in the first n components only contains Si literals, and the 4

This can be done by realizing that the theory is disconnected in the grounding for every pair (A, B) of individuals and applying the lifted case-analysis.

5

(n + 1)-th component contains no Si literals. These components are disconnected from each other, so we can consider each of them separately. The (n + 1)-th component comes from clauses in ? and is domain-liftable by definition. The following two Lemmas prove that the clauses in the other components are also domain-liftable. The proofs of both lemmas rely on domain recursion. Lemma 1. A clausal theory ?(S) with only one predicate S where all clauses have exactly two different literals of S is domain-liftable. Lemma 2. Suppose {?p1 , ?p2 , . . . , ?pn } are mutually exclusive subsets of ?x and {?q1 , ?q2 , . . . , ?qm } are mutually exclusive subsets of ?y . We can add any unit clause of the form ?pi ? ?pi , qj ? ?qj : S(pi , qj) or ?pi ? ?pi , qj ? ?qj : ?S(pi , qj) to the theory ?(S) in Lemma 1 and the theory is still domain-liftable. Therefore, theories in S 2 FO 2 and S 2 RU are domain-liftable. It can be easily verified that membership checking for S 2 FO 2 and S 2 RU is not harder than for FO 2 and RU , respectively. Example 3. Suppose we have a set ?j of jobs and a set ?v of volunteers. Every volunteer must be assigned to at most one job, and every job requires no more than one person. If the job involves working with gas, the assigned volunteer must be a non-smoker. And we know that smokers are most probably friends with each other. Then we will have the following first-order theory: ?v1 , v2 ? ?v , j ? ?j : ?Assigned(v1 , j) ? ?Assigned(v2 , j) ?v ? ?v , j1 , j2 ? ?j : ?Assigned(v, j1) ? ?Assigned(v, j2) ?v ? ?v , j ? ?j : InvolvesGas(j) ? Assigned(v, j) ? ?Smokes(v) ?v1 , v2 ? ?v : Aux(v1 , v2) ? (Smokes(v1) ? Friends(v1 , v2) ? Smokes(v2)) Predicate Aux is added to capture the probability assigned to the last rule (as in MLNs). This theory

is not in FO 2 , not in RU , and is not domain-liftable using R. However, the first two clauses are of the form described in Lemma 1, the third and fourth are in FO 2 (and also in RU), and the third clause, which contains Assigned(v, j), has no other PRVs with more than one LV. Therefore, this theory is in S 2 FO 2 (and also in S 2 RU) and domain-liftable based on Theorem 1. Example 4. Consider the birthday paradox introduced in Section 4. After Skolemization [30] for removing the existential quantifier, the theory contains ?p ? ?p , ?d ? ?d : S(p) ? ?Born(p, d), ?p ? ?p , d1 , d2 ? ?d : ?Born(p, d1) ? ?Born(p, d2), and ?p1 , p2 ? ?p , d ? ?d : ?Born(p1 , d) ? ?Born(p2 , d), where S is the Skolem predicate. This theory is not in FO 2 , not in RU , and is not domain-liftable using R. However, the last two clauses belong to clauses in Lemma 1, the first one is in FO 2 (and also in RU) and has no PRVs with more than one LV other than Born. Therefore, this theory is in S 2 FO 2 (and also in S 2 RU) and domain-liftable based on Theorem 1. Proposition 4. FO 2 ? RU , FO 2 ? S 2 FO 2 , FO 2 ? S 2 RU , RU ? S 2 RU , S 2 FO 2 ? S 2 RU . Proof. Let T ? FO 2 and T 0 be any of the theories resulting from exhaustively applying rules in R except lifted case analysis on T . If T initially contains a unary PRV with predicate S, either it is still unary in T 0 or lifted decomposition has replaced the LV with a constant. In the first case, we can follow a generic branch of lifted case-analysis on S, and in the second case, either T 0 is empty or all binary PRVs in T have become unary in T 0 due to applying the lifted decomposition and we can follow a generic branch of lifted case-analysis for any of these PRVs. The generic branch in both cases is in FO 2 and the same procedure can be followed until all theories become empty. If T initially contains only binary PRVs, lifted decomposition applies as the grounding of T is disconnected for each pair of individuals, and after lifted decomposition all PRVs have no LVs. Applying case analysis on all PRVs gives empty theories. Therefore, T ? RU . The theory ?x, y, z ? ?p : F(x, y) ? F(y, z) ? F(x, y, z) is an example of a RU theory that is not in FO 2 , showing RU 6? FO 2 . FO 2 and RU are special cases of S 2 FO 2 and S 2 RU respectively, where ? = ?, showing FO 2 ? S 2 FO 2 and RU ? S 2 RU . However, Example 3 is both in S 2 FO 2 and S 2 RU but is not in FO 2 and not in RU, showing S 2 FO 2 6? FO 2 and S 2 RU 6? RU . Since FO 2 ? RU and the class of added ?(S) clauses are the same, S 2 FO 2 ? S 2 RU . 6

1000
100
100
10
1 0.1
1
10
100
10 1
0.1
10
100

8

0.01

0.01 0.001

1

0.001

Domain Recursion

10 1 0.1

3

30

Population size WFOMC-v3.0

Domain Recursion

0.001

3000

Population size WFOMC-v3.0

(b)

(a)

300

0.01

Population size

WFOMC-v3.0

Time in seconds

1000

100 Time in seconds

Time in seconds

1000

Domain Recursion

(c)

Figure 1: Run-times for calculating the WFOMC of (a) the theory in Example 3, (b) the S4 clause, and (c) symmetric transitivity, using the WFOMC-v3.0 software (which only uses R) and comparing it to the case where we use the domain recursion rule, referred to as Domain Recursion in the diagrams.

6 Experiments and Results

In order to see the effect of using domain recursion in practice, we find the WFOMC of three theories with and without using the domain recursion rule: (a) the theory in Example 3, (b) the S4 clause, and (c) the symmetric-transitivity theory. We implemented the domain recursion rule in C++ and compiled the codes using the g++ compiler. We compare our results with the WFOMC-v3.0 software5 . Since this software requires domain-liftable input theories, for the first theory we grounded the jobs, for the second we grounded ?x , and for the third we grounded ?p . For each of these three theories, assuming —?x — = n for all LVs x in the theory, we varied n and plotted the run-time as a function of n. All experiments were done on a 2.8GH core with 4GB RAM under MacOSX. The run-times are reported in seconds. We allowed a maximum of 1000 seconds for each run. Obtained results can be viewed in Fig. 1. These results are consistent with our theory and indicate the clear advantage of using the domain recursion rule in practice. In Fig. 1(a), the slope of the diagram for domain

recursion is approximately 4 which indicates the degree of the polynomial for the time complexity. Similar analysis can be done for the results on the S4 clause and the symmetrictransitivity clauses represented in Fig. 1(b), (c). The slope of the diagram in these two diagrams is around 5 and 2 respectively, indicating that the time complexity for finding their WFOMC are n5 and n2 respectively, where n is the size of the population.

7

Discussion

We can categorize theories with respect to the domain recursion rule as: (1) theories proved to be domain-liftable using domain recursion (e.g., S4, symmetric transitivity, and theories in S 2 FO 2), (2) theories that are domain-liftable using domain recursion, but we have not identified them yet as such, and (3) theories that are not domain-liftable even when using domain recursion. We leave discovering and characterizing the theories in category 2 and 3 as future work. But here we show that even though the theories in category 3 are not domain-liftable using domain recursion, this rule may still result in exponential speedups for these theories. Consider the (non-symmetric) transitivity rule: ?x, y, z ? ?p : ?Friend(x, y) ? ?Friend(y, z) ? Friend(x, z). Since none of the rules in R apply to the above theory, the existing lifted inference engines ground ?p and calculate the weighted model count (WMC) of the ground theory. By grounding ?p , these engines lose great amounts of symmetry. Suppose ?p = {A, B, C} and assume we select Friend(A, B) and Friend(A, C) as the first two random variables for case-analysis. Due to the exchangeability of the individuals, the case where Friend(A, B) and Friend(A, C) are assigned to True and False respectively has the same WMC as the case where they are assigned to False and True. However, the current engines fail to exploit this symmetry as they consider grounded individuals non-exchangeable. By applying domain recursion to the above theory instead of fully grounding it, one can exploit the symmetries of the theory. Suppose ?p0 = ?p ? {P }. Then we can rewrite the theory as follows: ?y, z ? ?p0 : ?Friend(P, y) ? ?Friend(y, z) ? Friend(P, z) 5

Available at: https://dtai.cs.kuleuven.be/software/wfomc

7

?x, z ? ?p0 : ?Friend(x, P) ? ?Friend(P, z) ? Friend(x, z) ?x, y ? ?p0 : ?Friend(x, y) ? ?Friend(y, P) ? Friend(x, P) ?x, y, z ? ?p0 : ?Friend(x, y) ? ?Friend(y, z) ? Friend(x, z) Now if we apply lifted case analysis on Friend(P, y) (or equivalently on Friend(P, z)), we do not get back the same theory with reduced population and calculating the WFOMC is still exponential. However, we only generate one branch for the case where Friend(P, y) is True only once. This branch covers both the symmetric cases mentioned above. Exploiting these symmetries reduces the time-complexity exponentially. This suggests that for any given theory, when the rules in R are not applicable one may want to try the domain recursion rule before giving up and resorting to grounding a population.

8

Conclusion

We identified new classes of domain-liftable theories called S 2 FO 2 and S 2 RU by reviving the domain recursion rule. We also demonstrated how this rule

is useful for theories outside these classes. Our work opens up a future research direction for identifying and characterizing larger classes of theories that are domain-liftable using domain recursion. It also helps us get closer to finding a dichotomy between the theories that are domain-liftable and those that are not, similar to the dichotomy result of Dalvi and Suciu [6] for query answering in probabilistic databases. It has been shown [15, 16] that compiling the WFOMC rules into low-level programs (e.g., C++ programs) offers a (approx.) 175x speedup compared to other approaches. While compiling the previously known rules to low-level programs was straightforward, compiling the domain recursion rule to low-level programs without using recursion might be tricky as it relies on the population size of the logical variables. A future research direction would be finding if the domain recursion rule can be efficiently compiled into low-level programs, and measuring the amount of speedup it offers. Acknowledgements.

2   References

[1] Babak Ahmadi, Kristian Kersting, and Sriraam Natarajan. Lifted online training of relational models with stochastic gradient methods. In ECML PKDD, pages 585?600, 2012. [2] W. W. Rouse Ball. Other questions on probability. Mathematical Recreations and Essays, page 45, 1960. [3] Paul Beame, Guy Van den Broeck, Eric Gribkoff, and Dan Suciu. Symmetric weighted first-order model counting. In PODS, pages 313?328, 2015. [4] Hung Hai Bui, Tuyen N Huynh, Artificial Intelligence Center, and Sebastian Riedel. Automorphism groups of graphical models and lifted variational inference. In UAI, page 132, 2013. [5] Jaesik Choi, Rodrigo de Salvo Braz, and Hung H. Bui. Efficient methods for lifted inference with aggregate factors. In AAAI, 2011. [6] Nilesh Dalvi and Dan Suciu. Efficient query evaluation on probabilistic databases. The VLDB Journal, 16(4):523?544, 2007. [7] Luc De Raedt, Angelika Kimmig, and Hannu Toivonen. ProbLog: A probabilistic Prolog and its application in link discovery. In IJCAI, volume 7, 2007. [8] Luc De Raedt, Kristian Kersting, Sriraam Natarajan, and David Poole. Statistical relational artificial intelligence: Logic, probability, and computation. Synthesis Lectures on Artificial Intelligence and Machine Learning, 10(2):1?189, 2016. [9] Rodrigo de Salvo Braz, Eyal Amir, and Dan Roth. Lifted first-order probabilistic inference. In IJCAI, pages 1319?1325, 2005. [10] Lise Getoor and Ben Taskar. Introduction to statistical relational learning. MIT press, 2007. [11] Vibhav Gogate and Pedro Domingos. Probabilistic theorem proving. In UAI, pages 256?265, 2011.

8

[12] Manfred Jaeger. Relational Bayesian networks. In UAI. Morgan Kaufmann Publishers Inc., 1997. [13] Yacine Jernite, Alexander M Rush, and David Sontag. A fast variational approach for learning Markov random field language models. In ICML, 2015. [14] Abhay Jha, Vibhav Gogate, Alexandra Meliou, and Dan Suciu. Lifted inference seen from the other side: The tractable features.

In NIPS, pages 973?981, 2010. [15] Seyed Mehran Kazemi and David Poole. Knowledge compilation for lifted probabilistic inference: Compiling to a low-level language. In KR, 2016. [16] Seyed Mehran Kazemi and David Poole. Why is compiling lifted inference into a low-level language so effective? arXiv preprint arXiv:1606.04512, 2016. [17] Kristian Kersting, Babak Ahmadi, and Sriraam Natarajan. Counting belief propagation. In UAI, pages 277?284, 2009. [18] Daphne Koller and Nir Friedman. Probabilistic Graphical Models: Principles and Techniques. MIT Press, Cambridge, MA, 2009. [19] Timothy Kopp, Parag Singla, and Henry Kautz. Lifted symmetry detection and breaking for MAP inference. In NIPS, pages 1315?1323, 2015. [20] Brian Milch, Luke S. Zettlemoyer, Kristian Kersting, Michael Haimes, and Leslie Pack Kaelbling. Lifted probabilistic inference with counting formulae. In AAAI, pages 1062?1068, 2008. [21] Mathias Niepert. Markov chains on orbits of permutation groups. In UAI, 2012. [22] David Poole, Fahiem Bacchus, and Jacek Kisynski. Towards completely lifted search-based probabilistic inference. arXiv:1107.4035 [cs.AI], 2011. [23] David Poole. First-order probabilistic inference. In IJCAI, pages 985?991, 2003. [24] Matthew Richardson and Pedro Domingos. Markov logic networks. Machine Learning, 62:107?136, 2006. [25] Parag Singla and Pedro M Domingos. Lifted first-order belief propagation. In AAAI, volume 8, pages 1094?1099, 2008. [26] Dan Suciu, Dan Olteanu, Christopher R?, and Christoph Koch. Probabilistic databases. Synthesis Lectures on Data Management, 3(2):1?180, 2011. [27] Nima Taghipour, Daan Fierens, Guy Van den Broeck, Jesse Davis, and Hendrik Blockeel. Completeness results for lifted variable elimination. In AISTATS, pages 572?580, 2013. [28] Guy Van den Broeck, Nima Taghipour, Wannes Meert, Jesse Davis, and Luc De Raedt. Lifted probabilistic inference by first-order knowledge compilation. In IJCAI, pages 2178?2185, 2011. [29] Guy Van den Broeck, Arthur Choi, and Adnan Darwiche. Lifted relax, compensate and then recover: From approximate to exact lifted probabilistic inference. In UAI, 2012. [30] Guy Van den Broeck, Wannes Meert, and Adnan Darwiche. Skolemization for weighted first-order model counting. In KR, 2014. [31] Guy Van den Broeck. On the completeness of first-order knowledge compilation for lifted probabilistic inference. In NIPS, pages 1386?1394, 2011. [32] Jan Van Haaren, Guy Van den Broeck, Wannes Meert, and Jesse Davis. Lifted generative learning of Markov logic networks. Machine Learning, pages 1?29, 2015. [33] Deepak Venugopal and Vibhav Gogate. Evidence-based clustering for scalable inference in Markov logic. In ECML PKDD, pages 258?273, 2014. [34] Deepak Venugopal and Vibhav G Gogate. Scaling-up importance sampling for Markov logic networks. In NIPS, pages 2978?2986, 2014.

9