# Exploiting Determinism to Scale Relational Inference

**Mohamed-Hamza Ibrahim** and **Christopher Pal** and **Gilles Pesant**
Department of Computer and Software Engineering, École Polytechnique Montréal
2500, Chemin de Polytechnique, Universite de Montréal, Montréal, Québec, Canada

## Abstract

One key challenge in statistical relational learning (SRL) is scalable inference. Unfortunately, most real-world problems in SRL have expressive models that translate into large grounded networks, representing a bottleneck for any inference method and weakening its scalability. In this paper we introduce Preference Relaxation (PR), a two-stage strategy that uses the determinism present in the underlying model to improve the scalability of relational inference. The basic idea of PR is that if the underlying model involves mandatory (i.e. hard) constraints as well as preferences (i.e. soft constraints) then it is potentially wasteful to allocate memory for all constraints in advance when performing inference. To avoid this, PR starts by relaxing preferences and performing inference with hard constraints only. It then removes variables that violate hard constraints, thereby avoiding irrelevant computations involving preferences. In addition it uses the removed variables to enlarge the evidence database. This reduces the effective size of the grounded network. Our approach is general and can be applied to various inference methods in relational domains. Experiments on real-world applications show how PR substantially scales relational inference with a minor impact on accuracy.

## Introduction

Relational and deterministic dependencies in data have been addressed with the emergence of statistical relational learning (SRL) (Getoor and Taskar, 2007). Typically in SRL the theory is propositionalized to a grounded network wherein any probabilistic inference like MC-SAT or belief propagation can be applied. However this approach can be very time-consuming: the grounded network is typically large, which slows down inference, and this can be problematic for learning when using inference as a subroutine.

Many SRL models, if not most, feature a set of structural properties like determinism, sparseness, symmetry, redundancy, and type hierarchy. Markov Logic is a powerful SRL formalism that exhibits these properties, and a simple way to enhance the scalability of inference on the underlying SRL model such as those instantiated by Markov logic is to exploit such properties. For instance, Lifted Inference(de Salvo Braz, Amir, and Roth, 2005; Singla and Domingos, 2008; Ahmadi et al., 2013; Kersting, 2012) uses the symmetry present in the structure of the network to reduce its size. Lazy Inference (Poon, Domingos, and Sumner, 2008; Singla and Domingos, 2006b) uses sparseness to ground out the theory lazily (i.e., we do not need to instantiate atoms and clauses that have default values), which yields gains in memory and time. Coarse-to-Fine Inference (Kiddon and Domingos, 2011) uses an induced type hierarchy over a domain's objects to apply sequences of increasingly fine approximations of inference to control the trade-off between lifting and accuracy. Although determinism sometimes poses a significant challenge to probabilistic inference, it remains an intrinsic part of the structure of SRL models. One approach (Allen and Darwiche, 2003) based on recursive conditioning has been proposed to exploit determinism in order to speed up exact inference by using standard logical techniques such as unit propagation. But the approach does not scale to large real-world problems. More recently, an efficient algorithm (Papai, Singla, and Kautz, 2011) based on ideas from constraint programming has been introduced to use determinism in order to prune objects in the domain of atoms by enforcing generalized arc consistency on the set of hard constraints in the theory. For each hard constraint, the algorithm iteratively performs two main operations (join and project) and therefore its complexity (both space and time) is sensitive to the complexity of those operations that are in some cases exponential in the number of atoms. In our work we take advantage of determinism in the underlying model to improve the scalability of inference for large, real-world problems.

It is common for many real-world problems in SRL to have expressive models that combine deterministic and probabilistic dependencies. The former often appear in the form of mandatory (i.e. hard) constraints that must be satisfied in any world with a non-zero probability. The latter are typically formulated as preferences (i.e. soft constraints), and dissatisfying them is not impossible, but less probable. Thus, if a query atom $X$ which is involved with a set of constraints $\mathcal{C} = \{\mathcal{H}, \mathcal{S}\}$ (where $\mathcal{H}$ and $\mathcal{S}$ are its subsets of hard and soft constraints respectively) violates one of the hard constraints $h$ in $\mathcal{H}$, then its marginal probability will be zero, even if it satisfies its other hard and soft constraints in

$\mathcal{C} - \{h\}$. A variable violates a hard constraint if there is a truth value for that variable such that the constraint is violated in all possible worlds consistent with that truth value assignment to the variable and the evidence. Current inference approaches do not exploit the fact that there is no need to consider irrelevant computations (and memory usage) with soft constraints $\mathcal{S}$, since using only hard constraints $\mathcal{H}$ should be sufficient to compute that its marginal probability $P(X)$ is zero. In relational domains, where we have millions of query atoms each involved with thousands of constraints, such irrelevant computations greatly weaken the scalability of the inference, especially if most query atoms have a tendency to violate hard constraints. Inspired by the sparseness property of relational domains, the marginal probability of the vast majority of query atoms being true is frequently zero. Potentially this is due to the violation of hard constraints that have at least one false precondition query atom.

In this paper we propose Preference Relaxation (PR), a two-stage strategy that uses the determinism (i.e. hard constraints) present in the underlying model to improve the scalability of relational inference. The basic idea of PR is to take advantage of the fact that hard constraints are sufficient to extract zero marginal probabilities for the considerable number of query ground atoms that violate them, and this without ever having to waste computational time (and memory usage) with preferences. To diminish such irrelevant computational time and memory, in a first stage PR starts by relaxing preferences and performing inference with hard constraints only in order to obtain the zero marginals for the query ground atoms that violate the hard constraints. It then filters these query atoms (i.e. it removes them from the query set) and uses them to enlarge the evidence database. In a second stage preferences are reinstated and inference is performed on a grounded network that is constructed based on both a filtered query set and an expanded evidence database obtained in the first stage. PR substantially reduces the effective size of the constructed grounded network, potentially with a loss of accuracy.

In what follows we begin by giving some background before demonstrating our general PR approach and illustrating how it applies to both MC-SAT and Belief Propagation algorithms. Finally we present experimental evaluations, followed by our conclusions.

## Background

**Notation.** We use $f \in \mathcal{F}$ to denote a constraint (or ground clause) that is a disjunction of literals built from $\mathcal{X}$, where $\mathcal{X} = \{X_1, X_2, cldots, X_n\}$ is a set of $n$ Boolean random variables representing ground atoms. Both "+" and "−" will be used to denote the true and false appearance of the variables. A weight $w_i$ is attached to each constraint $f_i$. For each variable $X_j$, we use $\theta_{X_j} = [\theta^+_{X_j}, \theta^-_{X_j}]$ to denote its positive and negative marginal probabilities, respectively. We use $f^s_i$ (resp. $f^h_i$) to indicate that the constraint $f_i$ is soft (resp. hard); the soft and the hard constraints are included in the two sets $\mathcal{F}^s$ and $\mathcal{F}^h$ respectively. The set $\mathcal{X}_h$, and $\mathcal{X}_s$ correspond to the variables appearing in the scope of hard constraints $\mathcal{F}^h$ and soft constraints $\mathcal{F}^s$, respectively.

**Propositional grounding.** Propositional grounding is the process of replacing a first-order Knowledge Base (KB) by an equivalent propositional one. In finite domains, inference over a first-order KB can be performed by propositional grounding followed by satisfiability testing. But in order to apply a satisfiability solver we need to create a Boolean variable for every possible grounding of every predicate in the domain and a propositional clause for every grounding of every first-order clause. One of the most powerful representations used in this area is Markov logic networks (Richardson and Domingos, 2006).

**Markov Logic Network (MLN).** A Markov Logic Network is a set of pairs $(f_i, w_i)$, where $f_i$ is a first-order logic formula (or clause in CNF) and $w_i$ is a numerical weight. Together with a finite set of constants, it defines a Markov logic network that has a binary variable node for each ground atom, and a feature for each ground clause (i.e. constraint). The weights attached to constraints reflect their strength dependencies. Thus hard constraints have the largest weight ($w \to \infty$), in the sense that they must be satisfied. A Markov logic compactly represents the joint distribution over unknown variables $\{X_1, \cdots, X_n\}$ given the evidence variables $E$ as: $P(X_1, \cdots, X_n|E) = \mathcal{Z}_E^{-1} e^{\sum_i w_i \cdot f^h_i} \cdot e^{\sum_k w_k \cdot f^s_i}$, where $\mathcal{Z}_E$ is the normalizing constant. For any possible world, $X_1, \cdots, X_n$, to have a non-zero probability, all the hard constraints have to be satisfied, in which their (infinite) weights can be factorized out of both the denominator and numerator.

**Probabilistic Inference.** The object of the inference task in a MLN is to compute either the most probable explanation of the grounded network, or to compute the marginal probability of the non-evidence variables (the query) given others as evidence. MC-SAT (Poon and Domingos, 2006), a slice sampling MCMC inference and loopy Belief Propagation (BP) (Yedidia, Freeman, and Weiss, 2005), a message-passing inference, are two widely-used algorithms that provide approximate marginals.

## Scaling Up Relational Inference via PR

In graphical models that feature determinism, it is wasteful to allocate memory to all constraints in advance when performing inference. The PR framework first allocates memory to the hard constraints along with their corresponding variables. In domains where most variables are more likely to violate hard constraints, this can save memory and also speed up the computation since we do not allocate memory and perform computations for soft constraints that do not affect the marginal probabilities of some query variables.

### The PR Framework

**Definition 1.** *Let $\mathcal{F}$ be a set of constraints. Each constraint $f_i(X_1, ...X_l) \in \mathcal{F}$, where the $X_j$'s are either variables or functions, has an inference state. The default state of constraints is **awake**, meaning that the constraint will be considered when constructing the underlying grounded network for answering the queries (i.e. for inference). Otherwise the constraint is **relaxed**, meaning that it will be ignored. A variable is **awake** if it appears directly as an argument of an*

*awake constraint or in a function that is an argument of an awake constraint. Otherwise it is considered to be **relaxed**.*

Assume that $\mathcal{A}$ is an inference algorithm that we want to transform via PR to obtain inference algorithm $\hat{\mathcal{A}}$. We make the following assumptions about $\mathcal{A}$ so that our framework can be used: (1) the theory of $\mathcal{A}$ involves carrying out a propositionalization step, then calling a propositional inference algorithm; (2) The propositional inference algorithm used by $\mathcal{A}$ updates one variable at a time.

We now describe how to derive $\hat{\mathcal{A}}$ from $\mathcal{A}$. Algorithm 1 takes as input an underlying model of constraints $\mathcal{M}$ (e.g. Markov logic model that involves clauses with associated weights), query variables $\mathcal{X}$ whose marginals we want to compute, a pruning threshold $\gamma$, and an evidence database $\mathcal{DB}$ that involves variables we can condition. Its output are the marginal probabilities $\Theta$ of query variables $\mathcal{X}$. In the first stage we relax all soft constraints and allocate memory for the hard constraints that are not satisfied when all evidence variables are set to their fixed values. This consequently awakens the non-evidence variables that are involved in these hard constraints. The awake variables and awake hard constraints become the initial set of variables and constraints that are used to perform inference, computing marginal probabilities of awake query variables. Since many local-search-based inference algorithms (e.g., WalkSAT) as well as MCMC-based algorithms resort to randomization to choose the next variable to update, when relaxing the preferences we apply a smart randomization technique (Poon, Domingos, and Sumner, 2008) over awake variables only. After convergence, $\hat{\mathcal{A}}$ filters the set of awake query variables: those whose marginals are at most $\gamma$ are removed from $\mathcal{X}$ and are added to evidence database $\mathcal{DB}$ as false evidence, and those whose marginals are at least $1-\gamma$ are added as true evidence.

In the second stage $\hat{\mathcal{A}}$ awakens all preferences that were previously relaxed. It then constructs the grounded network based on both the enlarged evidence database $\mathcal{DB}^*$ and the reduced set of query variables $\mathcal{X}^{**}$. That is to say it allocates memory for awake constraints (both hard and soft) that are not satisfied in the new evidence database $\mathcal{DB}^*$ to answer query variables $\mathcal{X}^{**}$. Then it applies the appropriate probabilistic inference to compute the marginal probabilities of awake query variables $\mathcal{X}^{**}$. It is worth noting that although the query variables are awakened at some point during inference, ultimately all of their marginal probabilities are guaranteed to be computed, but some are computed in the first stage (i.e., filtered query variables) and the rest will be computed in the second stage.

The advantage of our PR strategy is three-fold. First we avoid unnecessary computations with soft constraints to obtain marginal probabilities of awake query variables that violate awake hard constraints. Second we reduce the set of query variables $\mathcal{X}$ by filtering out some awake query variables that violate hard constraints. Third we enlarge the evidence database by adding those filtered query variables, which reduces the effective size of the grounded network that will be constructed for inference in the second stage since the evidence variables are fixed to their truth values.

**Algorithm 1:** PR-based Inference algorithm $\hat{\mathcal{A}}$

**Input**: Evidence database ($\mathcal{DB}$), set of query variables ($\mathcal{X}$), underlying model ($\mathcal{M}$), pruning Threshold ($\gamma$).
**Output**: Marginals of query variables ($\Theta$).

`// Preference Relaxation (PR) step`
$\mathcal{F}^h \leftarrow$ *awake* hard constraints not satisfied by $\mathcal{DB}$;
$\mathcal{X}_h \leftarrow$ *awake* query variables in $\mathcal{X}$ that appear in $\mathcal{F}^h$;
$\mathcal{M}^h \leftarrow$ ConstructNetwork($\mathcal{X}_h, \mathcal{F}^h, \mathcal{DB}$);
$\Theta_{\mathcal{X}_h} \leftarrow$ Infer($\mathcal{X}_h, \mathcal{M}^h$);
`// Filtering query variables` $\mathcal{X}_h$
**for** each $X_j \in \mathcal{X}_h$ **do**
  **if** $\theta^+_{X_j} \leq \gamma$ **then**
    $\mathcal{DB} \leftarrow \mathcal{DB} \cup \{\neg X_j\}$;
    $\Theta_{\hat{\mathcal{X}}} \leftarrow \Theta_{\hat{\mathcal{X}}} \cup \left\{\theta^+_{X_j}\right\}$; $\mathcal{X} \leftarrow \mathcal{X} \setminus \{X_j\}$;
  **else if** $\theta^+_{X_j} \geq 1-\gamma$ **then**
    $\mathcal{DB} \leftarrow \mathcal{DB} \cup \{X_j\}$;
    $\Theta_{\hat{\mathcal{X}}} \leftarrow \Theta_{\hat{\mathcal{X}}} \cup \left\{\theta^+_{X_j}\right\}$; $\mathcal{X} \leftarrow \mathcal{X} \setminus \{X_j\}$;
  **end if**
**end for**
$\mathcal{DB}^* \leftarrow \mathcal{DB}$; `// Enlarge evidence database`
$\mathcal{X}^* \leftarrow \mathcal{X}$; `//Shrink query set`
`// Awake all relaxed preferences`
$\mathcal{F}^* \leftarrow$ *awake* hard & soft constraints not satisfied by $\mathcal{DB}^*$;
$\mathcal{X}^{**} \leftarrow$ *awake* query variables in $\mathcal{X}^*$ that appear in $\mathcal{F}^*$;
`//Construct the reduced grounded network`
$\mathcal{M}^* \leftarrow$ ConstructNetwork($\mathcal{X}^{**}, \mathcal{F}^*, \mathcal{DB}^*$);
$\Theta_{\mathcal{X}^{**}} \leftarrow$ Infer($\mathcal{X}^{**}, \mathcal{M}^*$);
$\Theta \leftarrow \Theta_{\mathcal{X}^{**}} \cup \Theta_{\hat{\mathcal{X}}}$;
**Return** $\Theta$;

## PR-based Relational Inference Algorithms

In relational domains the variables are ground atoms which are defined over Boolean domains $\mathcal{D} = \{+, -\}$. A ground atom is an evidence if its marginal $\theta^+ = 0$ (false evidence) or $\theta^- = 0$ (true evidence). The constraints are ground clauses. Thus the ground clause is *relaxed* if it is set to a non-default state and otherwise it is *awake*. A ground atom is awake if it is included by (at least) one awake ground clause, otherwise it is relaxed.

PR-based inference algorithm $\hat{\mathcal{A}}$ is general and can be combined with many relational inference algorithms such as MCMC sampling (e.g. Gibbs sampling, simulated tempering, etc.), MC-SAT, Walk-SAT, and Belief propagation (BP). Here we illustrate PR-based relational inference for Belief propagation and MC-SAT, and use the latter in our experiments.

**PR-BP** Given a factor graph $\mathcal{G}$, PR-BP performs three steps. The situation is depicted in Fig. 1(b). **Step (i) - Relaxing the Factor Graph:** We relax soft factors and construct the factor graph for factor nodes and variable nodes that represent the awake ground hard clauses and their awake ground atoms, respectively. Assuming there is no evidence, all awake variable nodes are queries. We simulate BP in-

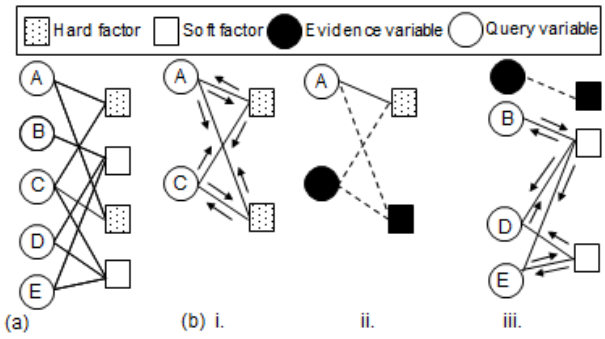


Figure 1: a) Propositional-BP. b) From left to right, the steps of PR-BP inference algorithm.

ference by alternating the passing of awake messages.[1] The message from an awake variable $X_j$ to an awake factor $f_i$ is:

$$\mu^{\text{awake}}_{X_j \to f_i} = \prod_{f_k \in \mathcal{F}_{X_j} \setminus \{f_i\}} \mu^{\text{awake}}_{f_k \to X_j} \quad (1)$$

and the message from an awake factor $f_i$ to awake variable $X_j$ is:

$$\mu^{\text{awake}}_{f_i \to X_j} = \sum_{\mathcal{X}_{f_i} \setminus \{X_j\}} \left( f_i^{\text{awake}} \prod_{X_k \in \mathcal{X}_{f_i} \setminus \{X_j\}} \mu^{\text{awake}}_{X_k \to f_i} \right) \quad (2)$$

**Step (ii) - Enlarging the Evidence Database**: We keep track of which variables and factors send and receive *violating awake messages*: an awake message is violating if it is less than $\gamma$ or greater that $1-\gamma$. The weight of an edge is the number of times the variable node receives an identical violating message from the same factor node. If we obtain a specified weight of such a message for a variable, then this means that it violates the hard factor. Thus we filter the variable and relax its messages. For instance at Fig. 1(b)-ii, if variable $C$ violates a factor we mark it as evidence (i.e. black circle) by adding it to $\mathcal{DB}$ and relax its passing of messages with this factor (i.e. dashed line). In addition if a factor is satisfied by marking one of its argument variable nodes as evidence then we relax this factor (i.e. black square) and relax its messages as well. **Step(iii) - Constructing the reduced Factor Graph:** We awake soft factors and construct the reduced grounded factor graph based on the last information we obtained from step (2). We then run the standard BP algorithm on it.

**PR-MC-SAT** PR-MC-SAT initializes by relaxing all soft clauses and awakens only those hard clauses that are not satisfied by the given evidence database $\mathcal{DB}$. It then calls Walk-SAT to find a solution to all awake hard clauses (which is a satisfying assignment to their awake ground atoms). At each iteration it generates membership $M$ by sampling from the currently satisfied awake hard clauses. Note that since the candidate clauses in $M$ are all hard, we do not select them based on their weights (all uniformly infinite) but whenever their auxiliary variable is greater than 1. We then run unit propagation in order to simplify the selected clauses in $M$. PR-MC-SAT next calls SampleSAT (Wei, Erenrich, and Selman, 2004) to obtain the next state over awake atoms by sampling near-uniformly from their solutions in $M$, which is initialized using smart randomization (Poon, Domingos, and Sumner, 2008) over the awake atoms. Once the $num\_samples$ threshold is reached we identify awake atoms whose marginals are less than $\gamma$ or greater than $1-\gamma$. These atoms are removed from the query set and added to the evidence database. We then awake the soft clauses and construct a grounded Markov network based on the shrunken query list and enlarged evidence database, on which we perform standard MC-SAT inference.

[1]The message is awake if it is passed between the awake variable node and the awake factor node, or vice versa.

## Combining PR with Lazy Inference

One key advantage of PR is that it can be combined with other state-of-the-art approaches which improve the scalability of inference, such as Lazy and Lifted. Algorithm 2 shows how to combine PR with Lazy MC-SAT (Poon, Domingos, and Sumner, 2008): it only differs from PR with propositional MC-SAT at Lines 2, 5, 6, and 8. Lazy-PR starts by calling Lazy-SAT to find the solution to awake hard constraints and then runs unit propagation among *active-awake* hard clauses in $M$ and their atoms. It then calls Lazy-SampleSAT to obtain the next state by sampling near-uniformly from the solutions in $M$. After reaching the $num\_samples$ threshold, it filters *active-awake* query atoms and enlarges the evidence database.

**Algorithm 2:** Combining PR with Lazy MC-SAT.

1: *Relax soft clauses and maintain only active ones hard clauses that are awakened;*
`// call Lazy-SAT`
2: $\mathcal{X}_h^{(0)} \leftarrow$ Satisfy (active-awake hard clauses);
3: **for** $i \leftarrow 1$ to $num\_samples$ **do**
4: $M \leftarrow \emptyset$;
5: **for** $\forall c_k \in$ active-awake hard clauses satisfied by $\mathcal{X}_h^{(i-1)}$ **do**
6: add $c_k$ to $M$;
7: **end for**
8: Sample $\mathcal{X}_h^{(i-1)} \sim \mathcal{U}_{\text{SAT(M)}}$; `//call Lazy SampleSAT`
9: **end for**
10: $[\mathcal{DB}^*, \mathcal{X}^*] \leftarrow$ *Filtering the active-awake query atoms* $\Theta_{\mathcal{X}_h}$;
11: *Construct the grounded network lazily on* $\mathcal{DB}^*$ *and* $\mathcal{X}^*$*;*
12: *call MC-SAT;*

# Experiments

The goal of our experimental evaluation is to investigate the following key questions. (**Q1**) Is PR powerful enough to reduce significantly the size of grounded networks? (**Q2**) Is PR-based inference competitive with prominent state-of-the-art methods such as Lazy Inference? (**Q3**) How is the scalability of PR-based inference influenced by the amount of determinism present in the underlying model and the pruning threshold that is used in filtering the query variables?

We experimented on a protein interaction task in a molecular biology domain, a link prediction task in a social networking domain, and an entity resolution task in a citation matching domain. In these problem domains it is possible to transform certain soft constraints into hard constraints. This provides us with the ability to test PR at differing levels of hard constraints.

To assess PR, we compared the space consumption and running time of construction and inference, and the accuracy (with and without PR) for both propositional MC-SAT (Poon and Domingos, 2006) and its lazy inference version (Poon, Domingos, and Sumner, 2008), as implemented in the Alchemy system (Kok et al., 2007). We conducted the experimental evaluations in the following manner. In the training phase, we learned the weights using a preconditioned scaled conjugate gradient (PSCG) algorithm (Lowd and Domingos, 2007) by performing a four-way cross-validation for protein interaction task, and a five-way cross-validation for both the link prediction and entity resolution tasks. In the testing phase, we carried out inference on the held-out dataset using six underlying inference algorithms (**MC-SAT, Lazy-MC-SAT, PR-MC-SAT 0.001, PR-MC-SAT 0.01, PR-Lazy-MC-SAT 0.001, PR-Lazy-MC-SAT 0.01**) to produce the marginals of all groundings of query atoms being true. We ran each algorithm until it either converged or its number of iterations exceeded 10 000. To obtain robust answers to the proposed questions, we did the following: 1) vary the number of objects in the domains, following methodology previously used (Poon, Domingos, and Sumner, 2008); 2) use two pruning thresholds $0.001$ and $0.01$ for PR; and 3) vary the amounts of determinism in the models (i.e., starting with hard constraints that initially exist in MLN, we gradually increase the number of hard constraints and re-run the experiment). All of the experiments were run on a cluster of nodes with $3.0$ GHz Intel CPUs, 3 GB of RAM, RED HAT Linux $5.5$, and we implemented PR as an extension to the Alchemy software (Kok et al., 2007).

## Protein Interaction

Here we use the MLN model of (Davis and Domingos, 2009) for a Yeast protein interaction problem. The Yeast problem captures information about a protein's location, function, phenotype, etc. The goal of inference is to predict the interaction relation (*Interaction, and Function*). Here, we ran the experiment at two amounts of determinism: 25% (i.e., ratio: 2 constraints out of 8 constraints, initially present as hard in MLN) and 37.5% (ratio: 3 constraints out of 8 constraints were considered hard), and we varied the number of objects in the domain from $0$ to $1000$ by increments of $50$.

## Link Prediction

For the link prediction task, we used the MLN model available on the Alchemy website (excluding the 22 unit clauses) of the UW-CSE dataset from (Richardson and Domingos, 2006). UW-CSE records information about the University of Washington (UW) Computer Science and Engineering Department (CSE). The inference task is to predict advisory relationships (*AdvisedBy*), and all other atoms are evidence (corresponding to all the information scenario (Richardson and Domingos, 2006)). Here we ran the experiment at two amounts of determinism: $\approx 9.7\%$ (ratio: 7 out of 72 constraints were hard) and $\approx 38.9\%$ (ratio: 28 out of 72 constraints were considered hard). In addition, we varied the number of objects in the domain from $0$ to $400$ by increments of $50$.

## Entity Resolution

For the entity resolution experiment, we used the MLN model that is similar to the established one of Singla (Singla and Domingos, 2006a) on the Cora dataset from (Poon, Domingos, and Sumner, 2008), consisting of $1295$ citations of $132$ different computer science papers. The goal of inference is to predict which pairs of citations refer to the same citation, author, title and venue (i.e, *SameBib, SameTitle, SameAuthor and SameVenue*). The other atoms are considered evidence atoms. We ran the experiment at two amounts of determinism: $\approx 12.5\%$ (ratio: 4 out of 32 constraints, actually already appearing in MLN as hard) and $\approx 25\%$ (ratio: 8 out of 32 constraints were considered hard). Additionally, we varied the number of objects in the domain from $0$ to $500$ by increments of $50$.

## Results

Figure 2 displays the time for total inference (where total inference = construction time + inference time) as a function of the number of objects in the domain for the six underlying inference algorithms at different amounts of determinism. The results show that PR-MC-SAT at thresholds 0.001 and 0.01 finishes at least four orders of magnitude faster than the propositional MC-SAT on both Yeast and UW-CSE datasets, and three orders of magnitude on the Cora dataset. It was also very competitive with Lazy Inference on Yeast and on UW-CSE with $38.9\%$ determinism. In addition, PR-lazy-MC-SAT at thresholds 0.001, 0.01 exceeds both the propositional MC-SAT and the lazy-MC-SAT on all underlying tested datasets. Clearly, PR-lazy-MC-SAT 0.01 was able to handle all full datasets, whereas lazy-MC-SAT ran out of memory with 1000 objects in the Yeast dataset.

Table 1 summarizes the average Construction (with and without PR) and inference times (mins.), memory (MB) and accuracy (CLL) metrics of Propositional grounding and PR-based MC-SAT inference algorithms on the underlying datasets. The results complement those of Figure 2, ensuring the promise of PR-based algorithms to improve MC-SAT's inference time and memory space. It also shows that RP-based algorithms at low thresholds (0.001) maintain approximately the same accuracy on the three underlying datasets, although they have a minor loss in accuracy to estimate marginals with a large pruning threshold (0.01), particularly in the Cora dataset.

## Discussion

Overall the results clearly show that PR-based algorithms substantially improve the scalability of propositional MC-SAT inference. This is due to, first, avoiding irrelevant computations as well as memory usage involving soft ground

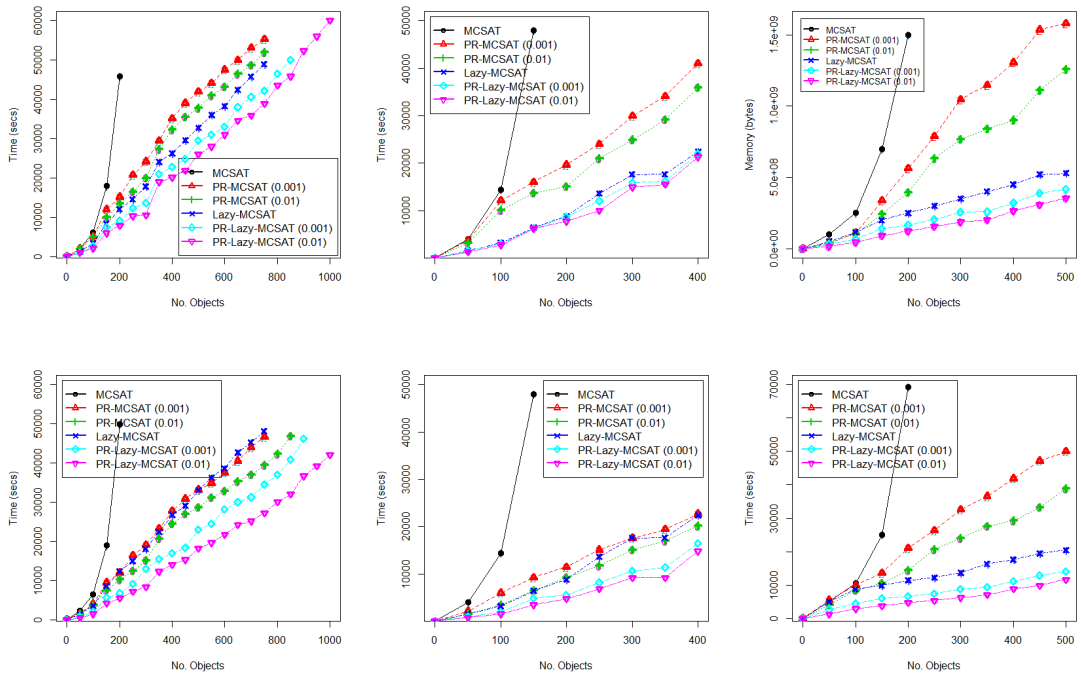


Figure 2: Inference time (secs) vs. number of objects in Yeast at 25%, 37.5% amounts of determinism (left), in UW-CSE at 9.7%, 38.9% amounts of determinism (middle), and in Cora at 12.5%, 25% amounts of determinism (right)

clauses for the large number of query ground atoms that violate hard ground clauses. Second, the enlarging of the evidence database that provides great implications for reducing the effective size of the grounded network.

Table 1: Average Construction (with and without PR) and Inference times (mins.), memory (MB) and accuracy (CLL) metrics of Propositional grounding and PR-based MC-SAT inference algorithms on the Yeast data set over 200 objects, the UW-CSE data set over 150 objects, and the Cora data set over 200 objects.

| Datasets | | Yeast | | UW-CSE | | Cora | |
|---|---|---|---|---|---|---|---|
| | | Determinism % | | Determinism % | | Determinism % | |
| Algo. | Metric | 25% | 37.5% | 9.7% | 38.9% | 12.5% | 25% |
| **Propositional** | *Cons.* | 63.82 | 63.83 | 7.92 | 7.92 | 80.85 | 81.01 |
| | *Infer.* | 716.26 | 797.44 | 783.75 | 873.38 | 1074.15 | 1390.33 |
| | *Mem.* | 389 | 429 | 1144.41 | 1445.93 | 1430.51 | 1653.87 |
| | *CLL* | −0.0353 | −0.0357 | −0.0433 | −0.0457 | −0.210 | −0.245 |
| **PR 0.001** | *PR-Cons.* | 99.93 | 117.58 | 13.27 | 14.97 | 162.5 | 170.5 |
| | *Infer.* | 150.94 | 90.15 | 169.05 | 140.03 | 270.13 | 249.62 |
| | *Mem.* | 125.85 | 93.29 | 314.68 | 213.60 | 535.88 | 439.43 |
| | *CLL* | −0.0354 | −0.0362 | −0.0443 | −0.0467 | −0.219 | −0.261 |
| **PR 0.01** | *PR-Cons.* | 98.33 | 110.82 | 12.98 | 13.09 | 195.7 | 201.02 |
| | *Infer.* | 126.32 | 70.38 | 114.06 | 93.60 | 164.3 | 131.51 |
| | *Mem.* | 112.40 | 80.17 | 258.39 | 160.93 | 376.97 | 301.58 |
| | *CLL* | −0.0365 | −0.0370 | −0.0459 | −0.0470 | −0.235 | −0.280 |

PR-based algorithms were also very competitive with respect to Lazy Inference whenever a substantial amount of determinism is present in the model. This can be attributed to the fact that determinism offers a trade-off for the capacity of PR-based algorithms in terms of saving/wasting computational time and memory: on one hand the PR step costs both memory and time for inference on a large number of hard clauses, but on the other hand it shrinks the query set and enlarges the evidence database. Moreover, PR-based algorithms dominate both propositional MC-SAT and lazy-MC-SAT on all tested data sets when they were combined with Lazy Inference. This is because the result of such combination is the exploitation of both sparseness in Lazy and determinism in PR to scale up the inference.

## Conclusion and Future work

We have proposed Preference Relaxation, a two-stage strategy that exploits determinism to scale up relational inference. Preferences are first ignored in order to shrink the query set and enlarge the evidence database. Experiments on real-world domains show that PR is able to greatly reduce the time and space requirements of inference by several orders of magnitude when applied to propositional MC-SAT and is twice as fast as its lazy version. In the future we plan to apply PR to other inference algorithms such as Lifted Inference. We also intend to perform an on-line inference scenario of PR-based algorithms.

## Acknowledgments

The authors would like to thank the anonymous reviewers for their constructive comments and suggestions. This work was funded by Natural Science and Engineering Research Council (NSERC) Discovery Grants Program and the Egyptian Cultural Affairs and Mission Sector. We sincerely thank these sponsors.

## References

Ahmadi, B.; Kersting, K.; Mladenov, M.; and Natarajan, S. 2013. Exploiting symmetries for scaling loopy belief propagation and relational training. *Machine Learning Journal* 92(1):91–132.

Allen, D., and Darwiche, A. 2003. New advances in inference by recursive conditioning. In *UAI*, 2–10. Morgan Kaufmann.

Davis, J., and Domingos, P. 2009. Deep transfer via second-order markov logic. In *Proceedings of the 26th International Conference on Machine Learning (ICML-09)*.

de Salvo Braz, R.; Amir, E.; and Roth, D. 2005. Lifted first-order probabilistic inference. In *International joint conference in artificial intelligent*.

Getoor, L., and Taskar, B. 2007. *Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning)*. The MIT Press.

Kersting, K. 2012. Lifted probabilistic inference. In *ECAI*, 33–38.

Kiddon, C., and Domingos, P. 2011. Coarse-to-fine inference and learning for first-order probabilistic models. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, California, USA, August 7-11, 2011*.

Kok, S.; Singla, P.; Richardson, M.; Domingos, P.; Sumner, M.; Poon, H.; and Lowd, D. 2007. The alchemy system for statistical relational ai. In *Technical report, Department of Computer Science and Engineering,University of Washington, Seattle, WA*. http://alchemy.cs.washington.edu.

Lowd, D., and Domingos, P. 2007. Efficient weight learning for markov logic networks. In *Knowledge Discovery in Databases: PKDD 2007, 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, Warsaw, Poland, September 17-21, 2007, Proceedings*, 200–211.

Papai, T.; Singla, P.; and Kautz, H. 2011. Constraint propagation for efficient inference in markov logic. In *Proceedings of 17th International Conference on Principles and Practice of Constraint Programming (CP 2011)*, number 6876 in Lecture Notes in Computer Science (LNCS), 691–705.

Poon, H., and Domingos, P. 2006. Sound and efficient inference with probabilistic and deterministic dependencies. In *Proceedings of the 21st national conference on Artificial intelligence - Volume 1*, AAAI'06, 458–463. AAAI Press.

Poon, H.; Domingos, P.; and Sumner, M. 2008. A general method for reducing the complexity of relational inference and its application to mcmc. In *AAAI*, 1075–1080. AAAI Press.

Richardson, M., and Domingos, P. 2006. Markov logic networks. *Mach. Learn.* 62(1-2):107–136.

Singla, P., and Domingos, P. 2006a. Entity resolution with markov logic. In *ICDM*, 572–582. IEEE Computer Society.

Singla, P., and Domingos, P. 2006b. Memory-efficient inference in relational domains. In *AAAI*. AAAI Press.

Singla, P., and Domingos, P. 2008. Lifted first-order belief propagation. In *AAAI*, 1094–1099. AAAI Press.

Wei, W.; Erenrich, J.; and Selman, B. 2004. Towards efficient sampling: Exploiting random walk strategies. In *AAAI*, 670–676. AAAI Press / The MIT Press.

Yedidia, J.; Freeman, W.; and Weiss, Y. 2005. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory* (7):2282–2312.