Non-Count Symmetries in Boolean & Multi-Valued Prob. Graphical Models

Ankit Anand[1]
Department of CSE
I.I.T Delhi

Ritesh Noothigattu[1]
Machine Learning Department[1]
Carnegie Mellon University

Parag Singla and **Mausam**
Department of CSE
I.I.T Delhi

Abstract

Lifted inference algorithms commonly exploit symmetries in a probabilistic graphical model (PGM) for efficient inference. However, existing algorithms for Boolean-valued domains can identify only those pairs of states as symmetric, in which the number of ones and zeros match exactly (*count symmetries*). Moreover, algorithms for lifted inference in multi-valued domains also compute a multi-valued extension of count symmetries only. These algorithms miss many symmetries in a domain.

In this paper, we present first algorithms to compute *non-count symmetries* in both Boolean-valued and multi-valued domains. Our methods can also find symmetries between multi-valued variables that have different domain cardinalities. The key insight in the algorithms is that they change the unit of symmetry computation from a variable to a variable-value (VV) pair. Our experiments find that exploiting these symmetries in MCMC can obtain substantial computational gains over existing algorithms.

1 Introduction

A popular approach for efficient inference in probabilistic graphical models (PGMs) is *lifted inference* (see [11]), which identifies repeated sub-structures (symmetries), and exploits them for computational gains. Lifted inference algorithms typically cluster symmetric states (variables) together and use these clusters to reduce computation, for example, by avoiding repeated computation for all members of a cluster via a single representative. Lifted versions of several inference algorithms have been developed such as variable elimination [21, 6], weighted model counting [8], knowledge compilation [26], belief propagation [23, 10, 24], variational inference [2], linear programming [19, 16] and Markov Chain Monte Carlo (MCMC) [28, 9, 17, 25, 1].

Unfortunately, to the best of our knowledge, all algorithms compute a limited notion of symmetries, which we call *count symmetries*. A count symmetry in a Boolean-valued domain is a symmetry between two states where the total number of zeros and ones exactly match. An illustrative algorithm for Boolean-valued PGMs (which we build upon) is Orbital MCMC [17]. It first uses graph isomorphism to compute symmetries and later uses these symmetries in an MCMC algorithm. Symmetries are represented via permutation groups in which variables interchange values to create other symmetric states. Notice, that if a state has k ones then any permutation of that state will also have k ones; this algorithm can only compute count symmetries.

Similarly, lifted inference algorithms for multi-valued PGMs (e.g., [21, 2]), only compute a weak extension of count symmetries for multi-valued domains – they allow symmetries only between those sets of variables that have the same domain. And, the count, i.e. the number of occurences, of any value (from the domain) within this set of variables remains the same between two symmetric states.

In response, we develop extensions to existing frameworks to enable computation of *non-count symmetries* in which the count of a value between symmetric states can change. We can also compute a special form of non-count symmetries, *non-equicardinal symmetries* in multi-valued domains, in which two variables that have different domain sizes may be symmetric. Our key insight is the framework of symmetry groups over variable-value (VV) pairs, instead of just variables. It allows interchanging a specific value of a variable with a different value of a different variable.

Orbital MCMC suffices for downstream inference over most kinds of symmetries except non-equicardinal ones, for which a Metropolis Hastings extension is needed. Our new symmetries lead to substantial computational gains over Orbital MCMC and vanilla Gibbs Sampling, which doesn't exploit any symmetries. We make the following

[1]First two authors contributed equally to the paper. Most of the work was done while the second author was at IIT Delhi.

Proceedings of the 20^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54.

contributions:

1. We develop a novel framework for symmetries between variable-value (VV) pairs, which generalize existing notions of variable symmetries (Section 3).
2. We develop an extension of this framework, which can also identify Non-Equicardinal (NEC) symmetries, i.e., among variables of different cardinalities (Section 4).
3. We design a Metropolis Hastings version of Orbital MCMC called NEC-Orbital MCMC to exploit NEC symmetries (Section 5).
4. We experimentally show that our proposed algorithms significantly outperform strong baseline algorithms (Section 6). We also release the code for wider use[2].

2 Background

Let $\mathcal{X} = \{X_1, X_2, \cdots, X_n\}$ denote a set of Boolean valued variables. A state $s = \{(X_i, v_i)\}_{i=1}^n$ is a complete assignment to variables in $\mathcal{X}$, with values $v_i \in \{0, 1\}$. We will use the symbol $\mathcal{S}$ to denote the entire state space.

A permutation θ of $\mathcal{X}$ is a bijection of the set $\mathcal{X}$ onto itself. $\theta(X_i)$ denotes the application of θ on the variable X_i. We will refer to θ as a *variable permutation*. A permutation θ applies on state s to produce $\theta(s)$, the state obtained by permuting the value of each variable X_i in s to that of $\theta(X_i)$. A set of permutations Θ is called a permutation group if it is closed under composition, contains the identity permutation, and each $\theta \in \Theta$ has its inverse in the set.

A graphical model $\mathcal{G}$ over the set of variables $\mathcal{X}$ is defined as the set of pairs $\{f_j, w_j\}_{j=1}^m$ where f_j is a feature function over a subset of variables in $\mathcal{X}$ and w_j is the corresponding weight [12]. Drawing parallels from automporphism of a graph where a variable permutation maps the graph back to itself, we define the notion of automorphism (referred to as symmetry, henceforth) of a graphical model as follows [18].

Definition 2.1. *A permutation θ of $\mathcal{X}$ is a* variable symmetry *of $\mathcal{G}$ if application of θ on $\mathcal{X}$ results back in G itself, i.e., the same set $\{f_j, w_j\}_{j=1}^m$ as in $\mathcal{G}$. We also call such permutations as variable permutations.*

Correspondingly, we define the autormorphism group of a graphical model.

Definition 2.2. *An* automorphism group *of a graphical model $\mathcal{G}$ is a permutation group Θ such that $\forall \theta \in \Theta$, θ is a variable symmetry of $\mathcal{G}$.*

Another important concept is the notion of an orbit of a state resulting from the application of a permutation group.

[2]https://github.com/dair-iitd/nc-mcmc

Definition 2.3. *The orbit (Γ) of a state s under the permutation group Θ, denoted by $\Gamma_\Theta(s)$, is the set of states resulting from application of permutations $\theta \in \Theta$ on s, i.e., $\Gamma_\Theta(s) = \{s' | \exists \theta \in \Theta, \theta(s) = s'\}$.*

Note that orbits form an equivalence partition of the entire state space. In this work, we are interested in orbits obtained by application of an automorphism group, because all states in such an orbit have the same joint probability. Let $P_\mathcal{G}(s)$ denote the joint probability of a state s under $\mathcal{G}$.

Theorem 2.1. *Let Θ be an automorphism group of $\mathcal{G}$. Then for all states s and permutations $\theta \in \Theta$: $P_\mathcal{G}(s) = P_\mathcal{G}(\theta(s))$.*

2.1 Graph Isomorphism for Computing Symmetries

The procedure for computing an automorphism group [17] first constructs a colored graph $G_V(\mathcal{G})$ from the graphical model $\mathcal{G}$, in which all features are clausal or all features are conjunctive.[3] In this graph there are two nodes for each variable, one for each literal, and a node for each feature in $\mathcal{G}$. There is an edge between two literal nodes of a variable, and between a literal node and a feature if that literal appears in that feature in the graphical model. Each node is assigned a color such that all 1 value nodes get the same color, all 0 value nodes get the same color (but different from 1 node color), and all feature nodes get a unique color based on their weight. That is, two feature nodes have the same color if their weights in $\mathcal{G}$ are the same.

A graph isomorphism solver (e.g., Saucy [5]) over $G_V(\mathcal{G})$ outputs the automorphism group of this graph through a set of permutations. These permutations can be easily converted to variable permutations of $\mathcal{G}$, because any output permutation always maps a variable's 0 and 1 nodes to another variable's 0 and 1 nodes, respectively. These permutations collectively represent an automorphism group of $\mathcal{G}$.

2.2 Orbital Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) methods are one of most popular methods for inference where exact inference is hard. In these methods, a Markov chain $\mathcal{M}$ is set up over the state space and samples are generated. Running the chain for a sufficiently long time, starts generating samples from the true distribution. Gibbs sampling is one of the simplest MCMC methods.

Orbital MCMC [17] adapts MCMC to use the given variable symmetries of the graphical model $\mathcal{G}$. Given a Markov Chain $\mathcal{M}$ and starting from state s_t, Orbital MCMC generates the next sample s_{t+1} in two steps:

- It first generates an intermediate state s'_t by sampling from the transition distribution of $\mathcal{M}$ starting from s_t

[3]Each model can be pre-converted to a new model in which all features are clausal.

- It then samples state s_{t+1} uniformly from $\Gamma_\Theta(s'_t)$, the orbit of s'_t

The Orbital MCMC chain so constructed converges to the same stationary distribution as original chain $\mathcal{M}$ and is proven to mix faster, because of the orbital moves.

3 Variable-Value (VV) Symmetries

Existing work has defined symmetries in terms of variable permutations. We observe that these can only represent orbits in which all states have exactly the same count of 0s and 1s. The simple reason is that any variable permutation only permutes the values in a state and hence the total count of each value remains the same. We name such type of symmetries as *count symmetries*.

We now give a formal definition of count symmetries for a general multi-valued graphical model, since our work applies equally to both Boolean-valued as well as any other discrete valued domains. Let $\mathcal{X} = \{X_1, X_2, \cdots, X_n\}$ denote a set of variables where each X_i takes values from a discrete valued domain D_i. A permutation θ of $\mathcal{X}$ is a *valid variable permutation* if it defines a mapping between variables having the same domain. Analogously, we define a *valid variable symmetry*. We will say that two domains D_i and D_j are *equicardinal* if $|D_i| = |D_j|$. We call such variables equicardinal variables.

Definition 3.1. *Given a set of variables $X \subseteq \mathcal{X}$ sharing the same domain D and a $v \in D$, $count_X(s, v)$ computes the number of variables in X taking the value v in state s.*

Definition 3.2. *Given a domain D, let X_D denote the subset of all the variables whose domain is D. A (valid) variable symmetry θ is a* count symmetry *if for each such subset $X_D \subseteq \mathcal{X}$, $count_{X_D}(s, v) = count_{X_D}(\theta(s), v)$, $\forall v \in D, \forall s \in \mathcal{S}$.*

Theorem 3.1. *For a graphical model $\mathcal{G}$, every (valid) variable symmetry θ is a count symmetry.*

We argue here that count symmetries are restrictive; a lot more symmetry can be exploited if we simultaneously look at the *values* taken by the variables in a state. To illustrate this, consider a very simple graphical model $\mathcal{G}_1$ with the following two formulas: (a) w_1: $a \vee \neg b$ (b) w_2: $\neg a \vee b$. It is easy to see that there is no non-trivial symmetry here. The permutation $\theta(a) = b, \theta(b) = a$ results in a different graphical model since the two formulas have different weights. On the other hand, if we somehow could permute a with $\neg b$ and b with $\neg a$, we would get back the same model. In this section, we will formalize this extended notion of symmetry which we refer to as *variable-value symmetry* (VV symmetry in short).

Definition 3.3. *Given a set of variables $\mathcal{X} = \{X_1, \cdots, X_n\}$ where each X_i takes values from a domain D_i, a* variable-value *(VV) set is a set of pairs $\{(X_i, v_l^i)\}$ such that each variable X_i appears exactly once with each $v_l^i \in D_i$ in this set where v_l^i denotes the l^{th} value in D_i. We will use $\mathcal{S}_\mathcal{X}$ to denote the VV set corresponding to $\mathcal{X}$.*

For example, given a set $\mathcal{X} = \{a, b\}$ of Boolean variables, the VV set is given by $\{(a, 0), (a, 1), (b, 0), (b, 1)\}$.

Definition 3.4. *A* Variable-Value permutation *ϕ over the VV set $\mathcal{S}_\mathcal{X}$ is a bijection from $\mathcal{S}_\mathcal{X}$ onto itself.*

Recall that a variable permutation applied to a state in a Boolean domain always results in a valid state. However, that may not be true in multi-valued domains, since if two variables that have different domains are permuted, it may not result in a valid state. It is also not true for all VV permutations. For example, given the state $[(a, 0), (b, 0)]$, a VV permutation defined as $\phi(a, 0) = (b, 1), \phi(a, 1) = (a, 1), \phi(b, 0) = (b, 0), \phi(b, 1) = (a, 0)$ results in the state [(b,1), (b,0)] which is inconsistent. Therefore, we need to impose a restriction on the set of allowed VV permutations so that they result in only valid states.

Definition 3.5. *We say that a VV permutation ϕ is a valid VV permutation if each variable $X_i \in \mathcal{X}$ maps to a unique variable X_j under ϕ. In other words, ϕ is valid if, whenever $\phi(X_i, v_l^i) = (X_j, v_{l'}^j)$ and $\phi(X_i, v_t^i) = (X_k, v_{t'}^k)$, then $X_j = X_k$, $\forall v_l^i, v_t^i \in D_i$. In such a scenario, we say that ϕ maps variable X_i to X_j.*

It is easy to see that for any valid VV permutation ϕ, applying ϕ on a state s always results in a valid state $\phi(s)$. It also follows that if such a ϕ maps a variable X_i to X_j, then D_i and D_j must be equicardinal.

Theorem 3.2. *The set of all valid VV permutations over $\mathcal{S}_\mathcal{X}$ forms a group.*

Consider a graphical model $\mathcal{G}$ specified as a set of pairs $\{f_j, w_j\}$. Each feature f_j can be thought of as a Boolean function over the variable assignments of the form $X_i = v_l^i$. Hence, action of a VV permutation ϕ on a feature f_j results in a new feature f'_j (with weight w_j) obtained by replacing the assignment $X_i = v_l^i$ by $X_j = v_{l'}^j$ in the underlying functional form of f_j where $\phi(X_i, v_l^i) = (X_j, v_{l'}^j)$. Hence, application of ϕ on a graphical model $\mathcal{G}$ results in a new graphical model $\mathcal{G}'$ where each feature (w_j, f_j) is transformed through application of ϕ. We are now ready to define the symmetry of a graphical model under the application of VV permutations.

Definition 3.6. *We say that a (valid) VV permutation is a* VV symmetry *of a graphical model $\mathcal{G}$ if application of ϕ on $\mathcal{G}$ results back in $\mathcal{G}$ itself.*

All other definitions of the previous section follow analogously. We can define an automorphism group over VV permutations, and also define an orbit of a state under this permutation group. VV symmetries strictly generalize the notion of variable symmetries.

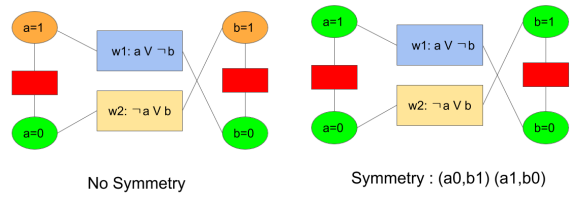

Figure 1: (a) Variable Symmetry Graph for toy example $\mathcal{G}_1$ (b) VV-Symmetry Graph for $\mathcal{G}_1$

Theorem 3.3. *Each (valid) variable symmetry θ can be represented as a VV symmetry ϕ. There exist valid VV symmetries that cannot be represented as a variable symmetry.*

Recall that a variable permutation θ is *valid* if it always maps between variables that have exactly the same domain. Say, $\theta(X_i) = X'_i$ with both variables having domains D_i. It is easy to see that ϕ defined such that $\phi(X_i, v_l^i) = (X'_i, v_l^i)$ for all $v_l^i \in D_i$, will result in the same sets of symmetric states.

To prove the second part, consider a PGM $\mathcal{G}_2$ with two Boolean variables X_1 and X_2. Let there be four features $f_{00}, f_{01}, f_{10}, f_{11}$, one corresponding to each of the four states, with weights given as w_s, w_d, w_d, w_s, respectively. Then, we have a VV symmetry ϕ such that $\phi(X_1, 0) = (X_2, 1)$, $\phi(X_1, 1) = (X_2, 0)$, $\phi(X_2, 0) = (X_1, 1)$ and $\phi(X_2, 1) = (X_1, 0)$. Note that ϕ maps the state $[(X_1, 0), (X_2, 0)]$ to $[(X_1, 1), (X_2, 1)]$ and reverse, and similarly there is a symmetry ϕ' which maps $[(X_1, 0), (X_2, 1)]$ to $[(X_1, 1), (X_2, 0)]$ and reverse. There is no variable symmetry which can capture the symmetries induced by ϕ since counts are not preserved. This proves the theorem. But let us for a moment define a renaming of the form $X'_1 = \neg X_1$. Variable symmetries will now be able to capture the symmetries due to ϕ but will miss out on the ones due to ϕ'. This is illustrative because there is no single problem formulation which can capture both the state symmetries above using the notion of variable symmetries alone.

Theorem 3.4. *VV symmetries preserve joint probabilities, i.e., for any VV symmetry ϕ, and state s: $P_{\mathcal{G}}(s) = P_{\mathcal{G}}(\phi(s))$.*

3.1 Computing Variable-Value Symmetries

We now adapt the procedure in Section 2.1 to compute VV symmetries in multi-valued domains. For a PGM $\mathcal{G}$ with clausal theory or conjunctive theory (as before), we construct a colored graph $G_{VV}(\mathcal{G})$ with a node for each variable-value pair. We also have a node for each feature, which is connected to the specific VV nodes it contains. We need to additionally impose a mutual exclusivity constraint to assert that a variable can only take exactly one of its many values. This is accomplished by adding exactly-one features with ∞ weight between all values of each variable. When assigning colors to each node, we assign all values

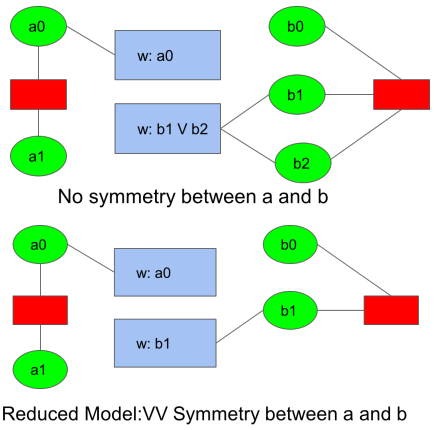

Figure 2: (a)Unreduced multi-valued domain $\mathcal{G}_3$ (b) Reduced multi-valued domain $\mathcal{G}_3$

of any variable the same color, as opposed to different values getting different colors. This allows the isomorphism solver to attempt discovering symmetries between different value nodes. As before, all features with the same weight get the same color. Figure 1 illustrates this on $\mathcal{G}_1$ where Variable symmetry assigns different colors to 0 and 1 while VV-Symmetry assigns a single color (green) to both 0 and 1 assignments of all variables.

We run Saucy [5] over $G_{VV}(\mathcal{G})$ to compute its automorphism group via a set of permutations. These permutations are valid VV-permutations (by construction of G_VV), and, collectively, represent a VV automorphism group of $\mathcal{G}$.

Theorem 3.5. *Any permutation ϕ that preserves graph isomorphism in $G_{VV}(\mathcal{G})$ is a valid VV-permutation for $\mathcal{G}$.*

Theorem 3.6. *The automorphism group of colored graph $G_{VV}(\mathcal{G})$ constructed above computes a VV-automorphism group of graphical model $\mathcal{G}$.*

4 Non-Equicardinal (NEC) Symmetries

While VV symmetries can compute non-count symmetries, they only consider mapping between equicardinal variables. In this section, we will deal with symmetries which can be present across variables having different domain sizes. Consider the following example graphical model $\mathcal{G}_3$ with two features: (1)w: $a = 1$ (2) w: $b = 1 \vee b = 2$. Let a and b have the domains D_a and D_b, respectively, specified as $D_a = \{0, 1\}$ and $D_b = \{0, 1, 2\}$. Clearly, there is no VV symmetry between a and b since they have different domain sizes. But intuitively, the two states given as $[(a, 1), (b, 0)]$ and $[(a, 0), (b, 1)]$ are symmetric to each other since in each case, exactly one of the two features having the same weight is satisfied. Similarly, for $[(a, 1), (b, 0)]$ and $[(a, 0), (b, 2)]$. Further, it is easy to see that the two values of $b = 1$ and $b = 2$ are symmetric to each other in the sense states of the form $[(a, v), (b, 1)]$ have the same probability as the states $[(a, v), (b, 2)]$ where $v \in \{0, 1\}$.

We will combine the above two ideas together to exploit symmetries using domain reduction. We first identify all the equivalent values of each variable and *replace* them by a single representative value. In this reduced graphical model, we then identify VV symmetries and finally translate them back to the original graphical model. In the following, we will assume that we are given a graphical model $\mathcal{G}$ defined over a set of n variables $\mathcal{X}$ where each $X_i \in \mathcal{X}$ takes values from a domain D_i. Further, we will use the symbol $\mathcal{D} = D_1 \times D_2, \cdots D_n$ to denote the cross product of the domains.

Definition 4.1. *Consider a variable $X_i \in \mathcal{X}$ and let $v, v' \in D_i$. Let $\phi^i_{v \leftrightarrow v'}$ denote a VV permutation which maps the VV pair (X_i, v) to (X_i, v') and back. For all the remaining VV pairs (X_k, v''), $\phi^i_{v \leftrightarrow v'}$ maps the pair back to itself. We refer to $\phi^i_{v \leftrightarrow v'}$ as a* value swap permutation *for variable X_i.*

In the example above, $\phi^b_{1 \leftrightarrow 2}$ is a value swap permutation for b which permutes the variable assignments $b = 1$ and $b = 2$, and keeps the remaining variable assignments, i.e., $b = 0$ and $a = 1$, fixed.

Definition 4.2. *A value swap permutation $\phi^i_{v \leftrightarrow v'}$ is a* value swap symmetry *of $\mathcal{G}$ if it maps $\mathcal{G}$ back to itself.*

In our running example, $\phi^b_{1 \leftrightarrow 2}$ is a value swap symmetry of $\mathcal{G}_4$. Next, we show that the set of all value swap symmetries corresponding to a variable X_i divides its domain D_i into equivalence classes.

Definition 4.3. *Given a graphical model $\mathcal{G}$, we define a relation SS_i (swap symmetry) over the set $D_i \times D_i$ as follows. Given $v, v' \in D_i$, $(v, v') \in SS_i$ if $\phi^i_{v \leftrightarrow v'}$ is a value swap symmetry of $\mathcal{G}$.*

It is easy to see that relation SS_i is an equivalence relation and hence, partitions the domain D_i into a set of equivalence classes. Given a value $v \in D_i$, we choose a representative value from its equivalence class based on some canonical ordering. We denote this value by $rep_i(v)$.

Next, we will define a reduced domain D^R_i obtained by considering one value from each equivalence set.

Definition 4.4. *Let SS_i divide the domain D_i into r equivalence classes. We define the reduced domain D^R_i as the r-sized set $\{v^*_j\}_{j=1}^r$ where v^*_j is the representative value for the j^{th} equivalence class. We will use $\mathcal{D}^R = D^R_1 \times D^R_2 \times \cdots D^R_n$ to denote the cross product of the reduced domains.*

Revisiting our example, the reduced domain for b is given as $D^R_b = \{0, 1\}$. Next we define a reduced graphical model $\mathcal{G}^R$ over the reduced set of domains $\{D^R_i\}_{i=1}^n$.

Definition 4.5. *Let $\mathcal{G}$ be a graphical model with the set of weighted features $\{w_j, f_j\}$. Let $X_i = v$ be a variable assignment appearing in the Boolean expression for f_j. We construct a new feature f'_j by replacing every such expression $X_i = v$ by $false$ (and further simplifying the expression) whenever $v \neq rep_i(v)$. If $v = rep_i(v)$, then we leave the assignment $X_i = v$ in f'_j as is. The reduced $\mathcal{G}^R$ is the graphical model having the set of features $\{w_j, f'_j\}$ defined over the set of variables $\mathcal{X}$ with X_i having the domain D^R_i.*

Intuitively, in $\mathcal{G}^R$, we restrict each variable X_i to take only the representative value from each of its equivalence classes. In our running example, the reduced graphical model is given as $\{w\colon a = 0; w\colon (b = 1) \vee false\}$ which is same as $\{w\colon a = 0; w\colon b = 1\}$. Since the domains have been reduced in $\mathcal{G}^R$, we may now be able to discover mappings which were not possible earlier. For instance in our running example, we now have a VV symmetry ϕ^R which maps $(a, 0)$ to $(b, 1)$ and back.

Let the joint distributions specified by $\mathcal{G}$ and $\mathcal{G}^R$ be given by P_G and P_{G^R}, respectively. The next theorem describes the relationship between these two distributions.

Theorem 4.1. *Let $\mathcal{G}$ be a graphical model and let $\mathcal{G}^R$ be the corresponding reduced graphical model. Consider a state s specified as $\{X_i, v_i\}_{i=1}^n$ where each $v_i \in D^R_i$. By definition, $v_i \in D_i$. We claim that $P_{\mathcal{G}^R}(s) = k * P_{\mathcal{G}}(s)$ where k is some constant $k \geq 1$ independent of the specific state s.*

Proof. Note that the reduced graphical model G^R is emulating the distribution specified by $\mathcal{G}$ where the space of possible variable assignments is now restricted to those belonging to the representative set, i.e., for each variable X_i the allowed set of values is now $D^R_i = \{v_i | v_i = rep_i(v), v \in D_i\}$. Therefore, G^R can be thought of as enforcing a conditional distribution over the underlying space given the fact that assignments can now only come from cross product set $\mathcal{D}^R$. Recall that state s is valid assignment in the original as well as the reduced graphical model. Therefore, we have $P_{\mathcal{G}^R}(s) = P_{\mathcal{G}}(s | s \in \mathcal{D}^R) = P_{\mathcal{G}}(s)/P_{\mathcal{G}}(s \in \mathcal{D}^R)$. Here, the denominator term $P_{\mathcal{G}}(s \in \mathcal{D}^R)$ is simply the probability that a randomly chosen state s in the original distribution belongs to the restricted domain set. Clearly, this is independent of the state s and let this given as $1/k$, where $k \geq 1$ is a constant independent of s. Then, $P_{\mathcal{G}^R}(s) = k * P_{\mathcal{G}}(s)$. □

Above theorem gives us a recipe to discover additional symmetries across variables having different domain sizes. Let $s = \{X_i, v_i\}_{i=1}^n$ be a state in $\mathcal{G}$. Let $rep(s)$ denote the representative state for s given as $\{(X_i, rep_i(v_i)\}_{i=1}^n)$.Following steps describe a procedure to get a new state s' symmetric to s using the idea of domain reduction.

Procedure NonEquiCardinalSym:

- Let $u = rep(s)$ denote the representative state for s.
- Apply a VV symmetry $\phi^R(u)$ over u in the reduced graphical model. Resulting state u' is symmetric to u in $\mathcal{G}^R$.

- Apply a series of n value swap symmetries of the form $\phi^i_{v'_i \leftrightarrow v''_i}$ over state u', one for each variable X_i such that $X_i = v'_i$ in u', $v''_i \in D_i$. Resulting state s' is symmetric to s in $\mathcal{G}$.

Definition 4.6. *Let τ be a permutation over the state space $\mathcal{S}$ of $\mathcal{G}$ defined using the Procedure NonEquiCardinalSym, i.e., $\tau(s) = \phi^n_{v'_n \leftrightarrow v''_n}(\phi^{n-1}_{v_{n-1} \leftrightarrow v''_{n-1}}(\cdots \phi^1_{v'_1 \leftrightarrow v''_1}(\phi^R(rep(s)))\cdots))$, where ϕ^R is a VV symmetry of $\mathcal{G}^R$ and each $\phi^i_{v'_i \leftrightarrow v''_i}$ is a value swap symmetry for variable X_i in $\mathcal{G}$. We refer to τ as a non-equicardinal symmetry of $\mathcal{G}$.*

Unlike VV symmetries whose action is defined over a VV pair, non-equicardinal symmetries directly operate over the state space. Their transformation of the underlying graphical model is implicit in the symmetries that compose them.

Theorem 4.2. *The set of all non-equicardinal symmetries forms a permutation group.*

Finally, we need to show that action of non-equicardinal symmetries indeed results in states which have the same probability.

Theorem 4.3. *Let τ be a non-equicardinal symmetry of a graphical model $\mathcal{G}$. Then, $P_\mathcal{G}(s) = P_\mathcal{G}(\tau(s))$.*

Proof. Let $s' = \tau(s)$. Let $u = rep(s)$. Since u' is obtained by application of VV symmetry $\phi^R(u)$ in $\mathcal{G}^R$, we have $P^R_\mathcal{G}(u) = P^R_\mathcal{G}(u')$. Using Theorem 4.1, this implies that $P_\mathcal{G}(u) = (1/k) * P_{\mathcal{G}^R}(u) = (1/k) * P_{\mathcal{G}^R}(u') = P_\mathcal{G}(u')$ for some constant k. Hence, u and u' have the same probability under $P_\mathcal{G}$.

Since $u = rep(s)$ can be obtained by application of n value swap symmetries over s (one for each variable), $P_\mathcal{G}(s) = P_\mathcal{G}(u)$. Similarly, since s' is obtained by an application of n value swap symmetries over u', we have $P_\mathcal{G}(u') = P_\mathcal{G}(s)$. Combining this with the fact that, $P_\mathcal{G}(u) = P_\mathcal{G}(u')$, we get $P_\mathcal{G}(s) = P_\mathcal{G}(s')$. □

4.1 Computing Non-Equicardinal Symmetries

We adapt the procedure in Section 3.1 by running graph isomorphism over a series of two colored graphs. Our first colored graph is constructed as in Section 3.1, except that all features are given different colors. This disallows any mapping between (X_i, v_i) and (X_j, v_j) for $X_i \neq X_j$, and only allows mapping between different values of a single variable. For example, in the running example, this would determine that $(b, 1)$ and $(b, 2)$ are symmetric. We then retain only the representative value for each equivalent set of VV pairs, and removes nodes and edges for other values.

We take this reduced colored graph and recolor all mutual exclusivity features with a single color. We run graph isomorphism again to obtain the VV symmetries of the reduced model. These permutations together with the single-variable permutations from the previous step gives the non-equicardinal symmetries of the original model.

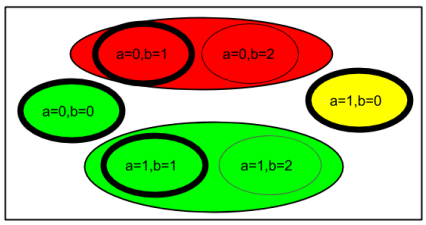

Figure 3: State Partition for Toy Example $\mathcal{G}_3$. Same Colored States are in same orbit. Large Ovals show sub-orbits and representative states of sub-orbits are with dark outline.

5 MCMC with VV & NEC Symmetries

Recall from Section 2.2 that variable symmetries are used in approximate inference via the Orbital MCMC algorithm. It alternates original MCMC move with an orbital move, which uniformly samples from the orbit of the current state. We first observe that the same algorithm will work for VV symmetries computed in Section 3.1, except that the orbital move will now sample from the orbit induced by VV permutations – we call this algorithm VV-Orbital MCMC.

We now consider the case of non-equicardinal symmetries in multi-valued PGMs. The main idea from Orbital MCMC remains valid – we need to alternate between original chain and orbital move. However, sampling a random state from an orbit is tricky now, because a non-equicardinal orbit may have a two-level hierarchical structure – it is an orbit over suborbits. The top level orbit is in the reduced model and is an orbit over representative states. At the bottom level, each representative state may represent multiple states via application of a *variable* number of value-swap symmetries.

As an example, consider the state partition in our running example, as illustrated in Figure 3. Each orbit is shown by a unique color, and suborbits by large ovals. The green orbit (top level) has two representative states (0,0) and (1,1) in the reduced model. If we make an orbital move in the reduced model, we can easily pick a representative state uniformly at random. However, the state (1,1) has a suborbit – it further represents two states in the original model, (1,1) and (1,2), via value-swap symmetries on variable b. Our sampling goal is to pick uniformly at random from an orbit in the *original* model, which means we need to pick a representative state in the reduced model proportional to the size of suborbit it represents. Once a suborbit is picked, we can easily pick a state uniformly at random from within it. To pick a representative proportional to the size of the suborbit, we use Metropolis Hastings in the reduced model – we name the resulting algorithm NEC-Orbital MCMC.

Let $c(s)$ represent the cardinality of the suborbit of state s, i.e., the number of states for which the representative state

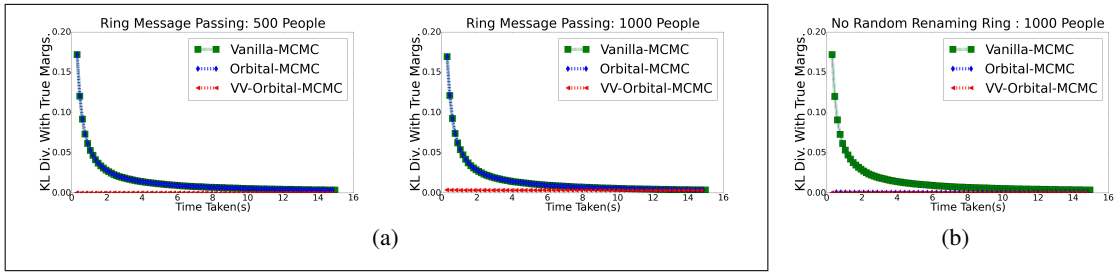

Figure 4: a)VV-Orbital-MCMC outperforms Orbital MCMC and Vanilla MCMC with different sizes of people on ring-message passing. b) VV-Orbital MCMC has negligible overhead compared to Orbital-MCMC

is the same as that of s: $|\{s'|rep(s') = rep(s)\}|$. Let $c^i(s)$ represent the number of states in the orbit of s which differ from s at most on the value of X_i , i.e., $|\{s'|rep(s') = rep(s), s.X_j = s'.X_j \forall j \neq i\}|$, where $s.X_j$ represents the value of X_j in s.

Given a Markov chain $\mathcal{M}$ over a graphical model $\mathcal{G}$, a sample from s_t to s_{t+1} in NEC-Orbital MCMC is generated:

- Generate s'_t by sampling from transition distribution of $\mathcal{M}$ starting from s_t.
- Let $u'_t = rep(s'_t)$. Sample u''_t (in $\mathcal{G}^R$) from the orbit $\Gamma_{\phi^R}(u'_t)$ via a Metropolis Hastings step using the uniform proposal distribution $q(\cdot) = \frac{1}{|\Gamma_{\Phi^R}(u'_t)|}$, and desired distribution $p(\cdot) \propto c(u''_t)$.
- Apply a series of n value swap symmetries of the form $\phi^i_{v''^i_t \leftrightarrow v^i_{t+1}}$ over state u''_t, one for each variable X_i, where $u''_t.X_i = v''^i_t$, and v^i_{t+1} is chosen uniformly at random from the set of values equivalent with v''^i_t with probability $\frac{1}{c^i(u''_t)}$. This is equivalent to sampling uniformly from the suborbit of u''_t.

Notice that sampling from the proposal distribution (uniform) from an orbit is easily accomplished by Product Replacement Algorithm [20]. MH accepts or rejects the sample with an Acceptance probability A, which can be computed by MH's detailed balance equation:

$$A(u'_t \rightarrow u''_t) = min\left(1, \frac{p(u''_t) * q(u'_t|u''_t)}{p(u'_t) * q(u''_t|u'_t)}\right) = min\left(1, \frac{p(u''_t)}{p(u'_t)}\right) = min\left(1, \frac{c(u''_t)}{c(u'_t)}\right)$$

The second equality above follows form the fact that $q(.)$ is a uniform proposal.

Theorem 5.1. *The Markov Chain constructed by NEC-Orbital MCMC converges to the unique stationary distribution of original markov chain $\mathcal{M}$.*

6 Experiments

We empirically evaluate our extensions of Orbital MCMC for both Boolean and multi-valued PGMs. In both settings, we compare against the baselines of vanilla MCMC, and Orbital MCMC [17]. In all orbital algorithms including ours, the base Markov chain $\mathcal{M}$ is set to Gibbs. We build our source code on existing code of Orbital MCMC.[4] It uses the Group Theory package Gap [7] for implementing the group-theoretic operations in the algorithms. We release our implementations for further research. [5] All our experiments are performed on Intel core i-7 machine. All our reported times include the time taken for computing symmetries.

Our experiments are aimed to assess the comparative value of our algorithms against baselines in those domains where a large number of symmetries (beyond count symmetries) are present. To this end, we construct two such domains. The first is a simple Boolean domain that shows how simple value renaming can affect baseline algorithms. The second is a multi-valued domain showcasing the potential benefits of non-equicardinal symmetries. The domains are:

Value-Renamed Ring Message Passing Domain: In this simple domain, N people with equal number of males and females are placed in a ring structure alternately with every male followed by a female, and they pass a bit of message to their immediate neighbor over a noisy channel. If X_i denoted the bit received by the i^{th} person, then we would have a formula for PGM $X_i \rightarrow X_{i+1}$ with weight w_1 if i is a male and weight w_2 if i is female. As a small modification to this domain, we randomly rename some X_is to mean $\neg$bit received by that agent, and change all formulas analogously. All the symmetries in the original ring should remain after this renaming. Our experiments test the degree to which the various algorithms are able to identify these.

Student-Curriculum Domain: In this multi-valued domain, there are K students taking courses from $|A|$ areas (e.g., theory, architecture, etc.). Each area $a \in A$ has a variable number of $N(a)$ courses numbered 1 to $N(a)$. Each student has to fulfill their breadth requirements by passing one course each from any two areas. A student has no specific preference to which of the $N(a)$ courses they take in an area. However, each student has a prior seriousness level, which determines whether they will pass

[4] https://code.google.com/archive/p/lifted-mcmc/

[5] Available at *https://github.com/dair-iitd/nc-mcmc*

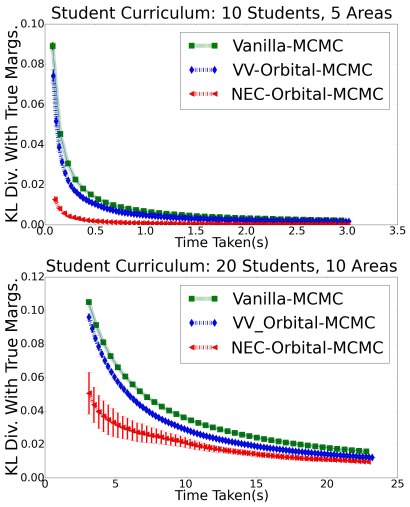

Figure 5: NEC-Orbital MCMC outperforms VV-Orbital MCMC and Vanilla-MCMC on student-curriculum domain.

any course. This scenario is modeled by defining a random variable P_{sa}, which is a multi-valued variable where value 0 denotes that student s failed the course in the area a, and value $i \in \{1 : N(a)\}$ denotes which course they passed. The weight for failing depends on the student but not on area. Finally, the variable $C_{saa'}$ denotes that s completed their requirements by passing courses from areas a and a'.

The Curriculum domain is interesting, because, for a given s, various values of P_{sa} other than 0 are all symmetric for all areas. And once, all P_{sa}s are converted to a representative value in the reduced model, all areas become symmetric for a student.

We compare different algorithms by plotting the KL-divergence of true marginals and an algorithm's marginals with time. True marginals are calculated by running Gibbs sampling for a sufficiently large duration of time. Figure 5 compares VV-Orbital MCMC with baselines on the message passing domain. The dramatic speedups obtained by VV-Orbital MCMC underscores Orbital MCMC's inability to identify the huge number of variable-renamed symmetries present in this domain, whereas VV-Orbital MCMC is able to benefit from these tremendously.

Before describing results on Curriculum domain, we first highlight that, out of the box, Orbital MCMC cannot run on this domain, because both its theory and implementation have only been developed for Boolean-valued PGMs. To meaningfully compare against Orbital MCMC, we first *binarize* the domain, by converting each multi-valued random variable P_{sa} into many Boolean variables P_{sac}, one for each value c. We need to add an infinite-weighted exactly-one constraint for each original variable before giving it to Orbital MCMC. A careful reader may observe that this binarization is already very similar to the VV construction of Section 3, but without non-equicardinal symmetries. Thus, this is already a much stronger baseline than currently found in literature.

Figure 5 shows the results on this domain. NEC-Orbital MCMC outperforms both baselines by wide margins. Orbital MCMC does improve upon vanilla Gibbs since it is able to find that all P_{sac}s for different cs are equivalent, however, it is unable to combine them across areas.

In domains where symmetries beyond count symmetries are not found, the overhead of our algorithms is not significant, and they perform almost as well as (binarized) Orbital MCMC (e.g., see Figure 4(b)). This is also corroborated by the fact that the time for finding symmetries is relatively small compared to the time taken for actual inference on both the domains. Specifically, this time is 0.250 sec and 0.009 sec for curriculum and ring domains, respectively.

In summary, both VV-Orbital MCMC and NEC-Orbital MCMC are useful advances over Orbital MCMC.

7 Conclusion and Future Directions

Existing lifted inference algorithms capture only a restricted set of symmetries, which we define as *count symmetries*. To the best of our knowledge, this is the first work that computes symmetries beyond count symmetries. To compute these *non-count symmetries*, we introduce the idea of computation over variable-value (VV) pairs. We develop a theory of VV automorphism groups, and provide an algorithm to compute these. These can compute equicardinal non-count symmetries, i.e., between variables that have the same cardinality. An extension to this allows us to also compute *non-equicardinal symmetries*. Finally, we provide MCMC procedures for using these computed symmetries for approximate inference. In particular, the algorithm to use non-equicardinal symmetries requires a novel Metropolis Hastings extension to existing Orbital MCMC. Experiments on two domains illustrate that exploiting these additional symmetries can provide a huge boost to convergence of MCMC algorithms.

We believe that many real world settings exhibit VV symmetries. For example, in the standard Potts model used in Computer Vision [12], the energy function depends on whether the two neighboring particles take the same value or not, and not on the specific values themselves (hence, 00 would be symmetric to 11). Exploring VV symmetries in the context of specific applications is an important direction for future research.

We will also work on extending the theoretical guarantees of variable symmetries [17] to VV symmetries. Several notions of symmetries already exist in the Constraint Satisfaction literature [3]. It will be interesting to see how our approach can be incorporated into the existing framework of symmetries in CSPs.

Acknowledgements

We thank Mathias Niepert for his help with the orbital-MCMC code. Ankit Anand is being supported by the TCS Research Scholars Program. Mausam is being supported by grants from Google and Bloomberg. Both Mausam and Parag Singla are being supported by the Visvesvaraya Young Faculty Fellowships by Govt. of India.

References

[1] Ankit Anand, Aditya Grover, Mausam, and Parag Singla. Contextual Symmetries in Probabilistic Graphical Models. In *IJCAI*, 2016.

[2] H. Bui, T. Huynh, and S. Riedel. Automorphism Groups of Graphical Models and Lifted Variational Inference. In *UAI*, 2013.

[3] David Cohen, Peter Jeavons, Christopher Jefferson, Karen E. Petrie, and Barbara M. Smith. Symmetry Definitions for Constraint Satisfaction Problems. *Constraints*, 11(2):115–137, 2006.

[4] James Crawford, Matthew Ginsberg, Eugene Luks, and Amitabha Roy. Symmetry-breaking predicates for search problems. *KR*, 96:148–159, 1996.

[5] Paul T Darga, Karem A Sakallah, and Igor L Markov. Faster symmetry discovery using sparsity of symmetries. In *Design Automation Conference*, 2008.

[6] R. de Salvo Braz, E. Amir, and D. Roth. Lifted First-Order Probabilistic Inference. In *IJCAI*, 2005.

[7] The GAP Group. *GAP – Groups, Algorithms, and Programming, Version 4.7.9*, 2015.

[8] V. Gogate and P. Domingos. Probabilisitic Theorem Proving. In *UAI*, 2011.

[9] V. Gogate, A. Jha, and D. Venugopal. Advances in Lifted Importance Sampling. In *AAAI*, 2012.

[10] K. Kersting, B. Ahmadi, and S. Natarajan. Counting Belief Propagation. In *UAI*, 2009.

[11] A. Kimmig, L. Mihalkova, and L. Getoor. Lifted Graphical Models: A Survey. *Machine Learning*, 99(1):1–45, 2015.

[12] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[13] Timothy Kopp, Parag Singla, and Henry Kautz. Lifted Symmetry Detection and Breaking for MAP Inference. In *NIPS*, 2015.

[14] H. Mittal, P. Goyal, V. Gogate, and P. Singla. New Rules for Domain Independent Lifted MAP Inference. In *Proc. of NIPS-14*, pages 649–657, 2014.

[15] M. Mladenov, B. Ahmadi, and K. Kersting. Lifted Linear Programming. In *AISTATS*, 2012.

[16] M. Mladenov, K. Kersting, and A. Globerson. Efficient Lifting of MAP LP Relaxations Using k-Locality. In *AISTATS*, 2014.

[17] Mathias Niepert. Markov Chains on Orbits of Permutation Groups. In *UAI*, 2012.

[18] Mathias Niepert. Symmetry-Aware Marginal Density Estimation. In *AAAI*, 2013.

[19] J. Noessner, M. Niepert, and H. Stuckenschmidt. RockIt: Exploiting Parallelism and Symmetry for MAP Inference in Statistical Relational Models. In *AAAI*, 2013.

[20] I. Pak. The Product Replacement Algorithm is Polynomial. In *Foundations of Computer Science*, 2000.

[21] D. Poole. First-Order Probabilistic Inference. In *IJCAI*, 2003.

[22] S. Sarkhel, D. Venugopal, P. Singla, and V. Gogate. Lifted MAP inference for Markov Logic Networks. In *AISTATS*, 2014.

[23] P. Singla and P. Domingos. Lifted First-Order Belief Propagation. In *AAAI*, 2008.

[24] P. Singla, A. Nath, and P. Domingos. Approximate Lifting Techniques for Belief Propagation. In *AAAI*, 2014.

[25] G. Van den Broeck and M. Niepert. Lifted probabilistic inference for asymmetric graphical models. In *AAAI*, 2015.

[26] G. Van den Broeck, N. Taghipour, W. Meert, J. Davis, and L. De Raedt. Lifted Probabilistic Inference by First-order Knowledge Compilation. In *IJCAI*, 2011.

[27] Guy Van den Broeck and Adnan Darwiche. On the complexity and approximation of binary evidence in lifted inference. In *NIPS*, 2013.

[28] D. Venugopal and V. Gogate. On Lifting the Gibbs Sampling Algorithm. In *NIPS*, 2012.