# Lifted Linear Programming

**Martin Mladenov** **Babak Ahmadi** **Kristian Kersting**
Knowledge Discovery Department, Fraunhofer IAIS
53754 Sankt Augustin, Germany
{firstname.lastname}@iais.fraunhofer.de

## Abstract

Lifted inference approaches have rendered large, previously intractable probabilistic inference problems quickly solvable by handling whole sets of indistinguishable objects together. Triggered by this success, we show that another important AI technique is liftable, too, namely linear programming. Intuitively, given a linear program (LP), we employ a lifted variant of Gaussian belief propagation (GaBP) to solve the systems of linear equations arising when running an interior-point method to solve the LP. However, this naïve solution cannot make use of standard solvers for linear equations and is doomed to construct lifted networks in each iteration of the interior-point method again, an operation that can itself be quite costly. To address both issues, we show how to read off an equivalent LP from the lifted GaBP computations that can be solved using any off-the-shelf LP solver. We prove the correctness of this compilation approac and experimentally demonstrate that it can greatly reduce the cost of solving LPs.

## 1 Introduction

Probabilistic logical languages, see [14, 11, 10] for overviews, provide powerful formalisms for knowledge representation and inference. They allow one to compactly represent complex relational and uncertain knowledge. For instance, in the friends-and-smokers Markov logic network (MLN) [30], the weighted formula 1.1 : $\mathtt{fr}(\mathtt{X},\mathtt{Y}) \Rightarrow (\mathtt{sm}(\mathtt{X}) \Leftrightarrow \mathtt{sm}(\mathtt{Y}))$ encodes that friends in a social network tend to have similar smoking habits. Yet performing inference in these languages is extremely costly, especially if it is done at the propositional level. Instantiating all atoms from the formulae in a such a model induces a standard graphical model with symmetric, repeated potential structures for all grounding combinations. Recent advances in lifted probabilistic inference such as [27, 12, 24, 33, 32, 8, 36, 16, 1] have rendered many of these large, previously intractable problems quickly solvable by exploiting the induced redundancies. For instance lifted belief propagation (BP) approaches [33, 19, 1] have been proven successful on several important AI tasks such as link prediction, social network analysis, satisfiability and boolean model counting problems. They automatically group nodes and potentials of the graphical model into supernodes and superpotentials if they have identical computation trees (i.e., the tree-structured "unrolling" of the graphical model computations rooted at the nodes). Lifted BP then runs a modified BP on this lifted (compressed) network.

Triggered by this success, in particular of lifted BP approaches, we show that another important AI technique is liftable, too, namely linear programming. Indeed, at the propositional level, considerable attention has been already paid to the link between BP and linear programming. This relation is natural since the MAP inference problem can be relaxed into linear programming, see e.g. [38]. At the lifted level, however, the link has not been established nor explored yet. Doing so significantly extends the scope of lifted inference since it paves the way to lifted solvers for linear assignment, allocation and flow problems as well as novel lifted (relaxed) solvers for SAT problems, Markov decision problems and maximum aposteriori (MAP) inference within probabilistic models, among others. To illustrate this, consider an extension of the friends-and-smokers MLN [30] to targeted advertisement [7]. Suppose we want to serve smoking-related advertisements selectively to the smoking users of a website and advertisements not related to smoking, e.g. sport

Appearing in Proceedings of the $15^{th}$ International Conference on Artificial Intelligence and Statistics (AISTATS) 2012, La Palma, Canary Islands. Volume 22 of JMLR: W&CP 22. 

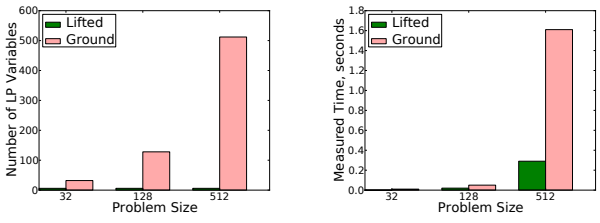



Figure 1: Computing an advertisement delivery schedule: Number of variables in the lifted and ground LPs **(left)** and measured time for solving the ground LP versus time for lifting and solving **(right)**.

ads, to the non-smoking users, as to maximize the expected number of advertisements that will be clicked on. Companies make considerable revenue through advertising, and consequently attracting advertisers has become an important and competitive endeavor. Assume that a particular delivery schedule for advertisements is defined by the matrix $\texttt{show}(\texttt{AType},\texttt{U}) \geq 0$ denoting the number of times that advertisement of type AType is to be shown on a web site to a particular user U, who may be a smoker with a certain probability, in a given period of time (e.g. a day). Assume further that we know the probability $\texttt{click}(\texttt{AType},\texttt{UType})$ that advertisement of AType will be clicked on if shown to a person type $\texttt{UType} \in \{\texttt{Sm},\texttt{NonSm}\}$. We model the overall probability of an advertisement of certain type to be clicked on by a given user, as the expectation $\texttt{click}(\texttt{AType},\texttt{U}) :=$

$$\sum\nolimits_{\texttt{UType}} \texttt{click}(\texttt{AType},\texttt{UType}) \cdot \mathit{prob}(\texttt{UType},\texttt{U}) \ .$$

where $\mathit{prob}(\texttt{UType},\texttt{U})$ is the probability that a user is a smoker obtained by running inference in the friends-and-smokers MLN. We can express the expected number of clicks for any schedule $\mathbf{X}$ as $\sum_{\texttt{AType}} \sum_{\texttt{U}} \mathit{prob}(\texttt{AType},\texttt{U}) \cdot \texttt{show}(\texttt{AType},\texttt{U})$. Our goal now is to find the schedule that maximizes this expectation. However, companies typically enter into contracts with advertisers and promise to deliver a certain number $\texttt{quota}(\texttt{AType})$ of advertisements of any type, $\sum_{\texttt{U}} \texttt{show}(\texttt{AType},\texttt{U}) \geq \texttt{quota}(\texttt{AType})$. Moreover, if a certain user visits the site only $\texttt{visits}(\texttt{U})$ times per day, our daily delivery schedule should not expect to serve more than $\texttt{visits}(\texttt{U})$ advertisements to them, $\sum_{\texttt{AType}} \texttt{show}(\texttt{AType},\texttt{U}) \leq \texttt{visits}(\texttt{U})$. Thus, we would like to find the schedule that maximizes the expected number of clicks with respect to these constraints. This is a linear program and, since we can exploit symmetries within the friends-smokers MLN, it is intuitive to expect that we can also do so for solving this linear program. As a sneak preview, we illustrate in Fig.1 that this is indeed the case – compression and efficiency gain are achieved when processing the linear program with our method. To show why and how this can be done is exactly the focus of the present paper. Specifically, our contribution is the first application of lifted inference techniques to linear programming.

To start, we note that the core computation of Bickson *et al.*'s [4] interior-point solver for LPs, namely solving systems of linear equations using Gaussian belief propagation (GaBP), can be naïvely lifted: replacing GaBP by Ahmadi *et al.*'s [1] lifted GaBP. In fact, this naïve approach may already results in considerable efficiency gains. However, we can do considerably better. The naïve solution cannot make use of standard solvers for linear equations and is doomed to construct lifted networks in each iteration of the interior-point method again, an operation that can itself be quite costly. To address both issues, we show how to read off an equivalent LP from the lifted GaBP computations. This LP can be solved using any off-the-shelf LP solver. We prove the correctness of this compilation approach, including a lifted duality theorem, and experimentally demonstrate that it can greatly reduce the cost of inference.

To do so, we proceed in three main steps. Step (S1) motivates our approach by briefly reviewing LPs and lifted GaBP and showing how to use lifted GaBP for solving LPs. Step (S2) argues that we can read off an equivalent system of linear equations, called lifted linear equations, from the computations of lifted GaBP, thus avoiding to run a modified (Ga)BP. Finally, step (S3) shows that this result can be extended to reading off a single LP only, the lifted LP, thus avoiding the time-consuming re-lifting in each iteration. Before concluding, we present our empirical evaluation on two different AI tasks.

## 2 (S1) Solving LPs by Lifted GaBP

A primal linear program $LP$ is a mathematical program of the following form:

$$\begin{aligned} \max_{\mathbf{x}} \quad & \mathbf{c}^T\mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b}, \quad \mathbf{x} \geq 0 \ , \end{aligned}$$

where $\mathbf{x} \in \mathbb{R}^n, \mathbf{c} \in \mathbb{R}^n, \mathbf{b} \in \mathbb{R}^m$ and $\mathbf{A} \in \mathbb{R}^{m \times n}$, $m \leq n$. We will also denote a LP in primal form as the tuple $LP = (A, b, c)$. Every primal LP has a dual linear program $\overline{LP}$ of the form

$$\min_{\mathbf{y}} \quad \mathbf{b}^T\mathbf{y} \ \text{ s.t. } \ \mathbf{A}^T\mathbf{y} \leq \mathbf{c} \ ,$$

where strong duality holds, namely, if $\mathbf{x}^*$ and $\mathbf{y}^*$ are optima of both $LP$ and $\overline{LP}$, $\mathbf{c}^T\mathbf{x}^* = \mathbf{b}^T\mathbf{y}^*$. A well-known approach for solving equality-constrained LPs, i.e., LPs in primal form is the primal barrier method, see e.g. [28], that is sketched in Alg. 1. It employs the Newton method to solve the following approximation

to the originial LP:

$$\begin{aligned} \max_{x,\mu} \quad & \mathbf{c}^T\mathbf{x} - \mu \sum_{k=1}^{n} \log x_k \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b}\,. \end{aligned}$$

At the heart of the Newton method lies the problem of finding an optimal search direction. This direction is the solution of the following set of linear equations:

$$\underbrace{\begin{bmatrix} -\mu\mathbf{X}^{-2} & \mathbf{A}^T \\ \mathbf{A} & 0 \end{bmatrix}}_{=:\mathbf{N}} \underbrace{\begin{bmatrix} \Delta\mathbf{x} \\ \lambda^+ \end{bmatrix}}_{=:\mathbf{d}} = \underbrace{\begin{bmatrix} \mathbf{c} + \mu\mathbf{X}^{-1}e \\ 0 \end{bmatrix}}_{=:\mathbf{f}}, \qquad (1)$$

where $\Delta\mathbf{x}$ is the Newton search direction, $\mathbf{X}$ is the diagonal matrix $\text{diag}(\mathbf{x})$ and $\lambda^+$ is a vector of Lagrangian multipliers that are discarded after solving the system.

Recently, Bickson *et al.* [4] have identified an important connection between barrier methods and probabilistic inference: solving the system of linear equations (1) can be seen as MAP inference within a pairwise markov random field (MRF) over Gaussians, solved efficiently using Gaussian Belief Propagation (GaBP). Specifically, suppose we want to solve a linear system of the form $\mathbf{N}\mathbf{d} = \mathbf{f}$ where we seek the column vector $\mathbf{d}$ such that the equality holds. Bickson et al. have shown how to translate this problem into a probabilistic inference problem. Given the matrix $\mathbf{N}$ and the observation matrix $\mathbf{f}$, the Gaussian density function $p(\mathbf{d}) \sim \exp(-\frac{1}{2}\mathbf{d}^t\mathbf{N}\mathbf{d} + \mathbf{f}^t\mathbf{d})$ can be factorized according to the graph consisting of edge potentials $\psi_{ij}$ and self potentials $\phi_i$ as follows: $p(\mathbf{d}) \propto \prod_{i=1}^{n} \phi_i(d_i) \prod_{i,j} \psi_{ij}(d_i, d_j)$, where the potentials are $\psi_{ij}(d_i, d_j) := \exp(-\frac{1}{2}d_i N_{ij} d_j)$ and $\phi_i(d_i) := \exp(-\frac{1}{2}N_{ii}d_i^2 + b_i d_i)$. The edge potentials $\psi_{ij}$ are specified for all $(i, j)$ s.t. $\mathbf{N}_{ij} > 0$. Computing the marginals for $d_i$ gives us the solution of $\mathbf{N}\mathbf{d} = \mathbf{f}$. As an illustration reconsider targeted advertisement from the previous section. Instantiating the problem for two people Alice and Bob gives us the following LP:

$$\begin{aligned} \max_{\mathbf{x}} \; & \sum_{U\in\{a,b\}} prob(\texttt{Sm},\texttt{U}) \cdot \texttt{click}(\texttt{SmAd},\texttt{U}) \\ & + prob(\texttt{NonSm},\texttt{U}) \cdot \texttt{click}(\texttt{SpAd},\texttt{U}) \qquad (2) \\ \text{s.t.} \; & \texttt{show}(\texttt{SmAd},\texttt{a}) + \texttt{show}(\texttt{SmAd},\texttt{b}) \geq \texttt{q}(\texttt{SmAd}), \\ & \texttt{show}(\texttt{SpAd},\texttt{a}) + \texttt{show}(\texttt{SpAd},\texttt{b}) \geq \texttt{q}(\texttt{SpAd}), \\ & \texttt{show}(\texttt{SmAd},\texttt{a}) + \texttt{show}(\texttt{SpAd},\texttt{a}) \leq \texttt{visits}(\texttt{a}), \\ & \texttt{show}(\texttt{SmAd},\texttt{b}) + \texttt{show}(\texttt{SpAd},\texttt{b}) \leq \texttt{visits}(\texttt{b}). \end{aligned}$$

Weiss *et al.*'s Gaussian belief propagation (GaBP) can be used for inference [37]. GaBP sends real-valued messages along the edges of the graph. Since $p(\mathbf{x})$ is jointly Gaussian, the messages are proportional to Gaussian distributions $\mathcal{N}(\mu_{ij}, P_{ij}^{-1})$ with precision $P_{ij} = -N_{ij}^2 P_{i\setminus j}^{-1}$ and mean $\mu_{ij} = -P_{ij}^{-1} N_{ij}\mu_{i\setminus j}$ where

$$P_{i\setminus j} = \tilde{P}_{ii} + \sum\nolimits_{k\in Nb(i)\setminus j} P_{ki}$$

**Algorithm 1:** Primal Barrier Algorithm

**Input**: $\mathbf{A}$, $\mathbf{b}$, $\mathbf{c}$, $\mathbf{x}^0$, $\mu^0$, $\gamma$, stopping criterion
**Output**: $\mathbf{x}^* = \arg\max_{\{\mathbf{x}|\mathbf{A}\mathbf{x}=\mathbf{b},\mathbf{x}\geq 0\}} \mathbf{c}^T x$

```
k ← 0;
while stopping criterion not fulfilled do
    Compute Newton direction Δx by solving (1);
    Set step size t by backtracking line search;
    Update x^{k+1} = x^k + t · Δx;
    Choose μ^{k+1} ∈ (0, μ^k);
    k ← k + 1
return x^k;
```

$$\mu_{i\setminus j} = P_{i\setminus j}^{-1}\big(\tilde{P}_{ii}\tilde{\mu}_{ii} + \sum\nolimits_{k\in Nb(i)\setminus j} P_{ki}\mu_{ki}\big)$$

for $i \neq j$ and $\tilde{P}_{ii} = N_{ii}$ and $\tilde{\mu}_{ii} = b_i/N_{ii}$. Here, $Nb(i)$ denotes the set of all the nodes neighboring the $i$th node and $Nb(i) \setminus j$ excludes the node $j$ from $Nb(i)$. All messages $P_{ij}$ are initially set to zero. The marginals are Gaussian probability density functions $\mathcal{N}(\mu_i, P_i^{-1})$ with precision $P_i = \tilde{P}_{ii} + \sum_{k\in Nb(i)} P_{ki}$ and mean $\mu_i = P_{i\setminus j}^{-1}\big(\tilde{P}_{ii}\tilde{\mu}_{ii} + \sum_{k\in Nb(i)} P_{ki}\mu_{ki}\big)$. If the spectral radius of the matrix $\mathbf{N}$ is smaller than 1 then GaBP converges to the true marginal means ($\mathbf{d} = \mu$). We refer to [3] for details. Although already quite efficient, many graphical models produce inference problems with symmetries not reflected in the graphical structure. Lifted BP variants can exploit this structure by automatically grouping nodes (potentials) of the graphical model $G$ into supernodes (superpotentials) — we denote by $g \sim g'$ that the nodes/factors $g$ and $g'$ have been compiled together into the same supernode/superfactor — if they have identical *computation trees* (i.e., the tree-structured unrolling of the graphical model computations rooted at the nodes). This compiled graph $\mathcal{G}$ is computed by passing around color signatures in the graph that encode the message history of each node. The algorithmic details of color passing are not important for this paper, we refer to [19]. The key point to observe is that the very same process also applies to GaBP (viewing "identical" for potentials only up to a finite precision), thus leading to the LGaBP algorithm introduced by Ahmadi *et al.* [1], which runs a modified GaBP on the lifted graph. The details of the modified GaBP are not important. We only note that messages now involve counts, denoted by $\sharp$, that essentially encode how often the message (potential) would have been used by GaBP on the original network $G$. For instance, $P_{i\setminus j}$ becomes now

$$P_{i\setminus j} = \tilde{P}_{ii} + \sum\nolimits_{k\in S(i)} \sharp_{ii}^k P_{ii}^k + \big(\sum\nolimits_{k\in Nb(i)\setminus i,j} \sharp_{ki} P_{ki}\big)$$

where $Nb(i)\setminus i, j$ denotes all neighbours of supernode $i$ without $i$ and $j$. The additional sum encodes the mes-

sages between different nodes of the same supernode $S(i)$. For details, we refer to [1]. Consider the following variant of (2) introducing symmetries by adding one more person into the domain:

$$\begin{aligned}
\max_{\mathbf{x}} & \sum_{U\in\{a,b,c\}} prob(\texttt{Sm},\texttt{U})\cdot\texttt{click}(\texttt{SmAd},\texttt{U}) \\
& + prob(\texttt{NonSm},\texttt{U})\cdot\texttt{click}(\texttt{SpAd},\texttt{U}) \qquad (3)\\
\text{s.t.} & \sum_{\texttt{p}\in\{\texttt{a},\texttt{b},\texttt{c}\}} \texttt{show}(\texttt{SmAd},\texttt{p}) \geq \texttt{q}(\texttt{SmAd}),\\
\text{s.t.} & \sum_{\texttt{p}\in\{\texttt{a},\texttt{b},\texttt{c}\}} \texttt{show}(\texttt{SpAd},\texttt{p}) \geq \texttt{q}(\texttt{SpAd}),\\
& \texttt{show}(\texttt{SmAd},\texttt{a}) + \texttt{show}(\texttt{SpAd},\texttt{a}) \leq \texttt{visits}(\texttt{a}),\\
& \texttt{show}(\texttt{SmAd},\texttt{b}) + \texttt{show}(\texttt{SpAd},\texttt{b}) \leq \texttt{visits}(\texttt{b}),\\
& \texttt{show}(\texttt{SmAd},\texttt{c}) + \texttt{show}(\texttt{SpAd},\texttt{c}) \leq \texttt{visits}(\texttt{c}).
\end{aligned}$$

Suppose we know as in the previous example that Alice is a smoker. For the others, however, we have no evidence. Thus they have identical cost in the objective and, due to the symmetric constraints, they are equal at the optimum. This is exactly what can be exploited by LGaBP.

Fig. 2 shows two ways to compute the solution of (1) in a single step of the log barrier method for the linear program given above. The first way (shown with dotted lines) constitutes of converting the matrix $N$ and the vector $f$ of our linear system (shown upper-left) into the corresponding MRF (shown below it). This MRF is lifted by color passing and by computing the marginals using LGaBP we obtain the solution of the linear system. From the picture it can be seen that lifting would group together the nodes corresponding to those who are indistinguishable in the MLN. The second way, outlined with solid lines, and its benefits are explained in the following.

## 3 (S2) Lifted Linear Equations

Recall that the modified GaBP sends messages involving counts. This suggests that we are actually running inference on a misspecified probabilistic model. We now argue that this is actually the case. Specifically, we show that the lifted problem can be compiled into an equivalent propositional problem of the same size. The resulting set of lifted linear equations can be solved using any standard solver, including GaBP. To see this, we start by noting that LGaBP computes a lifted, i.e., compiled version $\mathbf{r}\in\mathbb{R}^k$ of the solution vector $\mathbf{x}$ with $k\leq n$. The ground solution can be recovered as $\mathbf{x}=\mathbf{B}\mathbf{r}$ where $\mathbf{B}$ encodes the partition induced by the $\sim$ relation due to the color passing. It can be read off from the lifted graph $\mathcal{G}$ as follows: $B_{ij}=1$ if $x_i$ belongs to the $j$-th supernode of $\mathcal{G}$, and $B_{ij}=0$ otherwise. The matrix $\mathbf{B}$ has full column rank. Thus, when solving $\mathbf{A}\mathbf{x}=\mathbf{b}$ using LGaBP, we find $\mathbf{r}$ such that $\mathbf{A}\mathbf{B}\mathbf{r}=\mathbf{b}$ . Multiplying by the left pseudoinverse

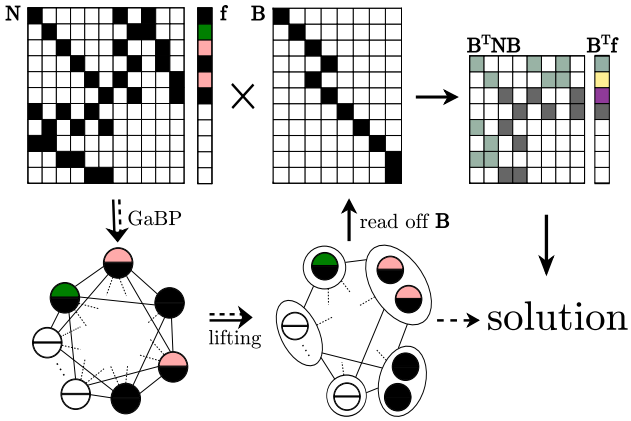



Figure 2: Lifted solving of the targeted advertisement LP with three persons. The dotted lines trace the LGaBP solution, whereas the solid lines follow the procedure in (S2). (best viewed in color)

of $\mathbf{B}$, i.e., $(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T$ on both sides, one obtains

$$(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{A}\mathbf{B}\mathbf{r}=(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{b}\,. \qquad (4)$$

The matrix $\mathbf{Q}:=(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{A}\mathbf{B}$ is the well-known quotient matrix [18] of the partition $\mathbf{B}$ of $\mathbf{A}$. Interestingly, a similar idea has been used to optimize Pagerank computations [2]. We state without proof that for the case of GaBP models, the partitioning of the nodes by color-passing corresponds to the so-called coarsest equitable partition of the graph. A defining characteristic of equitable partitions is that the following holds [17]:

$$\mathbf{A}\mathbf{B}=\mathbf{B}\mathbf{Q}\,. \qquad (5)$$

Thus, $\mathbf{Q}$ is invertible if $\mathbf{A}$ is invertible. To see this, let $\mathbf{u}$ be an eigenvector of $\mathbf{Q}$, i.e., $\mathbf{Q}\mathbf{u}=\lambda\mathbf{u}$. Multiplying from the left by $\mathbf{B}$ gives $\mathbf{B}\mathbf{Q}\mathbf{u}=\lambda\mathbf{B}\mathbf{u}$. Plugging in (5), this can be rewritten to $\mathbf{A}\mathbf{B}\mathbf{u}=\lambda\mathbf{B}\mathbf{u}$ . Now, if $\mathbf{A}$ is invertible, all its eigenvalues are non-zero. By the above the eigenvalues $\lambda$ of $\mathbf{Q}$ are also non-zero; $\mathbf{Q}$ is invertible.

Since we only deal with invertible matrices, both $\mathbf{A}\mathbf{x}=\mathbf{b}$ and (4) have a unique solution, and when solving (4) to obtain $\mathbf{r}$, then $\mathbf{x}=\mathbf{B}\mathbf{r}$ is the solution of $\mathbf{A}\mathbf{x}=\mathbf{b}$. Since $\mathbf{Q}\in\mathbb{R}^{k\times k}$, one obtains a problem of size equal to the size of the lifted LGaBP graph. Finally we note that $\mathbf{B}^T\mathbf{B}$ is a diagonal matrix where each entry on the diagonal is the (strictly positive) count of the respective supernode. Thus, we can directly compute $(\mathbf{B}^T\mathbf{B})^{-1}$ and may also solve

$$\mathbf{B}^T\mathbf{A}\mathbf{B}\mathbf{r}=\mathbf{B}^T\mathbf{b}. \qquad (6)$$

instead of solving (4). In other words, instead of lifting a solver for sets of linear equations, we lift the equations themselves and employ any standard solver.

Reconsider the targeted advertisement of the previous section and the solution of the linear system using LGaBP. Applying (6) to the problem of computing the Newton step for this program yields the second path of Fig. 2.

## 4 (S3.1) Lifted Linear Programming

Are there also lifted linear programs? That is, can we even avoid computing a set of lifted equations in each iteration of the barrier method and instead automatically compile the original LP into an equivalent LP that can be significantly smaller? In this section, we show that this is the case. For instance, (3) can automatically be compiled into (2).

Consider the linear system

$$\mathbf{A}_0\mathbf{v} = \mathbf{c}_0 \tag{7}$$

with $\mathbf{A}_0 = \begin{bmatrix} \mathbf{I}_n & \mathbf{A}^T \\ \mathbf{A} & \mathbf{I}_m \end{bmatrix}$ and $\mathbf{c}_0 = \begin{bmatrix} \mathbf{c} \\ \mathbf{b} \end{bmatrix}$, where $\mathbf{I}_n$ and $\mathbf{I}_m$ are identity matrices of size $n$ and $m$. We call this system the skeleton of the Newton search direction equation (1). The following Lemma says that the $\sim$ relation induced on $\mathbf{v}$ when solving the skeleton using LGaBP, denoted as $\sim_s$, is valid for solving (1) in all iterations of the barrier method applied: if $v_i \sim_s v_j$ then $\mathbf{x}_i^k = \mathbf{x}_j^k$ for $k = 0, 1, 2, 3, \ldots$ .

**Lemma 4.1** *Let $\mathbf{x}^0$ be an interior point of a linear program LP such that $x_i^0 = x_j^0$ if $v_i \sim_s v_j$. Then for all iterations $k$ of the barrier method it holds that $x_i^k = x_j^k$.*

**Proof** We are proving this in two steps. First, we prove that if $i \sim_s j$ for two variables $i$ and $j$ then $i \sim_b j$ for any $\sim$ relation produced by running the barrier method using LGaBP. Assume we are running the barrier method solving (1) using LGaBP. The MRFs constructed for all iterations $k$ differ only in the self potentials $\phi$, which in turn are completely specified by the $\mathbf{X} = \mathrm{diag}(\mathbf{x}) =: \mathbf{n}$ and $\mathbf{c} + \mu\mathbf{X}^{-1}e =: \mathbf{m}$ vectors. The edge potentials are not changing over the iterations of the barrier method. Consequently, the vectors $\mathbf{n}$ and $\mathbf{m}$ also determine the lifting produced in each iteration $k$ since they encode the color signatures of the nodes after one iteration of the color passing. So, as long as $n_i = n_j$ respectively $m_i = m_j$ for all $i$ and $j$ with $i \sim_b j$, color-passing will result in a $\sim$ relation that respects $i \sim_b j$, i.e., if $i \sim j$ then $i \sim_b j$. By design, however, this holds for the $\sim_s$.

Now, in the second part, we prove that using $\sim_s$ produces the same solution vector. We prove this by induction on $k$. For $k = 0$, this holds by choice of the initial feasible point. For $k \mapsto k+1$, consider two nodes $i$ and $j$ with $i \sim_s j$. First, $c_i = c_j$ by construction of the skeleton. Second, $x_i^k = x_j^k$ is true due to the induction hypothesis. Thus, $\hat{c}_i = c_i + \mu\frac{1}{x_i^k} = c_j + \mu\frac{1}{x_j^k} = \hat{c}_j$. This proves the induction step for the $\mathbf{c} + \mu\mathbf{X}^{-1}e$ values. In a similar way we can prove this for the $-\mu\mathbf{X}^{-2}$ values. Thus, the search direction vector $\Delta x$ respects the partition induced by $\sim_s$. □

Due to the lemma, we have identified a partitioning that is loop invariant of Alg. 1. We now show that a subset of it can be directly applied to the matrix $\mathbf{A}$ and the vector $\mathbf{b}$ of a primal LP.

We have already established that instead of running LGaB we may also simply solve (6) using any standard solver for systems of linear equations. When doing so, the solution of the original problem is given as $\mathbf{x} = \mathbf{B}\mathbf{r}$. We now have to impose the following restriction: when building the lifted graph of LGaBP, we leave the first $n$ variables ungrouped, even if the lifting tells us some of them should be grouped together. Clearly, this might result in loss of efficiency, although it does not affect the correctness of the LGaBP result. However, as we show below, it helps to derive a "relaxed" version of (6), which is of particular interest to us since it reveals how a lifted linear program can be constructed. The restriction can be translated to the equations by writing the block matrix $\mathbf{B}$ as

$$\mathbf{B} = \begin{bmatrix} \mathbf{I} & 0 \\ 0 & \mathbf{B}_M \end{bmatrix}, \tag{8}$$

so that the result of the LGaBP computation is expressed as

$$\begin{bmatrix} \Delta\mathbf{x} \\ \lambda^+ \end{bmatrix} = \begin{bmatrix} \mathbf{I} & 0 \\ 0 & \mathbf{B}_M \end{bmatrix} \mathbf{r}. \tag{9}$$

Now we argue in a similar way as in the previous section: if $\mathbf{r}$ is the lifted solution to (1), then $\mathbf{B}^T\mathbf{N}\mathbf{B}\mathbf{r} = \mathbf{B}^T\mathbf{f}$, or

$$\left(\mathbf{B}^T \begin{bmatrix} -\mu\mathbf{X}^{-2} & \mathbf{A}^T \\ \mathbf{A} & 0 \end{bmatrix} \mathbf{B}\right) \mathbf{r} = \mathbf{B}^T \begin{bmatrix} \mathbf{c} + \mu\mathbf{X}^{-1}e \\ 0 \end{bmatrix}.$$

With (8) we can compute this system explicitly as

$$\begin{bmatrix} -\mu\mathbf{X}^{-2} & \mathbf{A}^T\mathbf{B}_M \\ \mathbf{B}_M^T\mathbf{A} & 0 \end{bmatrix} \mathbf{r} = \begin{bmatrix} \mathbf{c} + \mu\mathbf{X}^{-1}\mathbf{e} \\ 0 \end{bmatrix}. \tag{10}$$

Note that $\mathbf{B}^T\mathbf{N}\mathbf{B}$ is invertible since $\mathbf{A}$ has full row-rank and the rows of $\mathbf{B}_M^T\mathbf{A}$ are sums of disjoint sets of the columns of $\mathbf{A}$, thus $\mathbf{B_M}^T\mathbf{A}$ also has full row rank. This means that the unique solution of (10) is a solution of (1), even though $\mathbf{B}$ does not neccessarily correspond to an equitable partition of $\mathbf{N}$.

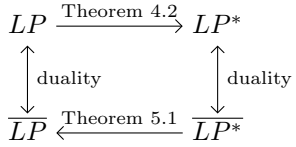



Figure 3: Links between LPs and lifted LPs.

Now, consider the $LP^* = (\mathbf{B}_M^T\mathbf{A}, \mathbf{c}, \mathbf{B}_M^T\mathbf{b})$ (note, the fact that we compile the $\mathbf{b}$-vector reveals why it is present in the skeleton equation). We prove the following lifting theorem for primal LPs.

**Theorem 4.2** *For every linear program $LP = (\mathbf{A}, \mathbf{b}, \mathbf{c})$, there exists a linear program $LP^* = (\mathbf{B}_M^T\mathbf{A}, \mathbf{B}_M^T\mathbf{b}, \mathbf{c})$ of smaller or equal size such that (1) the feasible region of $LP^*$ is a superset of the feasible region of $LP$, (2) $LP^*$ has at least one solution in common with $LP$, and (3) a common optimum to both will be found by Alg. 1 given a suitable initial point.*

**Proof** Let $\mathbf{x_0}$ be a feasible solution of $LP$, i.e. $\mathbf{A}\mathbf{x}_0 = \mathbf{b}$. Mupltiplying $\mathbf{B}_M^T$ on both sides preserves equality. Thus, $\mathbf{B}_M^T\mathbf{A}\mathbf{x}_0 = \mathbf{B}_M^T\mathbf{b}$. Therefore $\mathbf{x}_0$ is feasible for $LP^*$. This proves (1). Now, given an initial interior point that preserves the lifting of the skeleton, Eq. (1) for $LP^*$ is equivalent to (10) for $LP$ in every step of the primal barrier method due to Lemma 4.1. That is, solving one step for $LP^*$ is equivalent to solving one step of $LP$ using LGaBP since it obtains the same search direction $\Delta\mathbf{x}$ (Eq. (9)). This proves (2). Equality of both objective values also holds since the objective functions of the two LPs are the same. This proves (3). □

However, we have to be a little bit more careful. So far, we have assumed the existence of an intial points that preserves symmetries, i.e., the $\sim$ relation. We now justify this assumption.

Following Dantzig [9], we can always construct a modified version of $LP$, called by $LP_a$, by adding an extra variable $x_a$ associated with a very high cost $R$:

$$\begin{aligned} \max_{\mathbf{x},x_a} \quad & \mathbf{c}^T\mathbf{x} - Rx_a \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} + (\mathbf{b} - \mathbf{A}\mathbf{x}^+)x_a = \mathbf{b}, \\ & \mathbf{x} \geq 0, x_a \geq 0, \end{aligned}$$

where $x^+$ is any vector with positive components. This LP has the following properties, see also [9]: (A) if $R$ is sufficiently high, then the set of optimal solutions of $LP_a$ is the same as for $LP$, and (B) $\mathbf{x} = \mathbf{x}^+, x_a = 1$ is a valid feasible solution. Thus, one can choose $x^+ = \mathbf{1}$ which respects the symmetries of (7) for the original $LP$. Moreover, it can be shown that (7) for $LP_a$ has the same symmetries as for $LP$ except that the extra variable $x_a$ will not be grouped with any other variable.

**Algorithm 2:** Lifted Linear Programming

**Input**: An inequality-constrained LP ($\mathbf{A}$, $\mathbf{b}$, $\mathbf{c}$)
**Output**: $\mathbf{x}^* = \text{argmin}_{\{\mathbf{x}|\mathbf{A}\mathbf{x}\leq\mathbf{b}\}}\, \mathbf{c}^T\mathbf{x}$

1 Construct the equality-constrained LP $(\mathbf{A}^T, \mathbf{c}, \mathbf{b})$;
2 Lift the corresponding skeleton equation (7) using color-passing;
3 Read off the block matrix $\mathbf{B}_M$;
4 Obtain the solution $\mathbf{r}$ of the LP $(\mathbf{A}\mathbf{B}_M, \mathbf{b}, \mathbf{B}^T\mathbf{c})$ using any standard LP solver;
5 **return** $\mathbf{x}^* = \mathbf{B}\mathbf{r}$;

## 5 (S3.2) Lifted Duality

Theorem 4.2 shows that for every linear program, we can construct a possibly smaller linear program which has the property that when solved by the primal barrier method, or for that matter any other that maintains symmetries of this kind, will "simulate", at least partially, the use of lifted inference for the linear system solution step, so that relifting in every iteration can be avoided. The drawback of this method is that since the feasible region of the new LP may be larger than that of the original, it cannot be guaranteed that if a solution is found under different conditions (e.g., different solver or a non-symmetric interior point), it will still be valid for the original. As we show now, this situation is remedied when working with inequality constrained LPs. So, how do the lifted versions of LPs with inequality constraints look like? An elegant way to see this is to consider the dual linear program $\overline{LP^*}$ of the lifted program $LP^*$:

$$\min_{\mathbf{w}} \quad (\mathbf{B}_M^T\mathbf{b})^T\mathbf{w} \;\text{ s.t. }\; (\mathbf{B}_M^T\mathbf{A})^T\mathbf{w} \leq \mathbf{c}\,.$$

We show now that $\overline{LP^*}$ is the lifted version of $\overline{LP}$ and if $\mathbf{w}$ is a solution of $\overline{LP^*}$ then $\mathbf{y} = \mathbf{B}_M\mathbf{w}$ is a solution to $\overline{LP}$. In other words, we show a lifting theorem of duality:

**Theorem 5.1 (Lifted Duality)** *For every dual linear program $\overline{LP} = (\mathbf{A}^T, \mathbf{c}, \mathbf{b})$ there exists an equivalent dual linear program $\overline{LP^*} = (\mathbf{A}^T\mathbf{B}_M, \mathbf{c}, \mathbf{B}_M^T\mathbf{b})$ of smaller or equal size, whose feasible region and optima can be mapped to the feasible region and optima of $\overline{LP}$.*

**Proof** If $\mathbf{w}$ is a feasible solution of $\overline{LP^*}$ then $\mathbf{c} \geq (\mathbf{B}_M^T\mathbf{A})^T\mathbf{w} = \mathbf{A}^T(\mathbf{B}_M\mathbf{w}) = \mathbf{A}^T\mathbf{y}$. Thus, $\mathbf{y} = \mathbf{B}_M\mathbf{w}$ is a feasible solution of $\overline{LP}$. Moreover, any optimum of $\overline{LP^*}$ is an optimum of $\overline{LP}$. To see this, let $\mathbf{x}, \mathbf{x}^*, \mathbf{y}, \mathbf{w}$ be any optima of $LP, LP^*, \overline{LP}$ and $\overline{LP^*}$ respectively. By strong duality it holds that $\mathbf{c}^T\mathbf{x} = \mathbf{b}^T\mathbf{y}$ and $\mathbf{c}^T\mathbf{x}^* =$

$(\mathbf{B}_M^T\mathbf{b})^T\mathbf{w}$. By Theorem 4.2 we have $\mathbf{c}^T\mathbf{x} = \mathbf{c}^T\mathbf{x}^*$. Therefore, $\mathbf{b}^T\mathbf{y} = (\mathbf{B}_M^T\mathbf{b})^T w$. □

Thus, if we want to lift an inequality constrained linear program, we can simply view it as the dual of some primal program and apply Theorem 5.1. More importantly, however, the theorem tells us that the feasible region of the lifted dual is a subset of the feasable region of the dual such that it countains at least one optimum of the original problem. Thus, inequality-constrained LPs can be solved by any LP solver as we do not have to worry about initial points. This is summarized in Alg. 2, which we call lifted linear programming since using Theorems 4.2 and 5.1 the agorithm can be applied to any form of LPs, cf. Fig. 3.

## 6 Illustrative Evaluation

Our intention here is to investigate the following questions: **(Q1)** Are there LPs that can be solved more efficiently by lifting? **(Q2)** How much can we gain, given that we sacrifice the coarsest lifting for the construction of the lifted program. **(Q3)** How does lifting relate to the sparse vs. dense paradigm. Is it only making use of the sparsity in the LPs?

To this aim, we implemented lifted linear programming within Python[1] calling CVXOPT[2] as LP solver. All experiments were conducted on a standard Linux desktop computer.

**(Q1) Lifted MAP inference:** As shown in previous works, inference in graphical models can be dramatically sped-up using lifted inference. Furthermore, a relaxed version of MAP inference can be solved by linear programs using the well-known LP relaxation, see e.g. [15] for details. Thus, it is natural to expect that the symmetries in graphical models which can be exploited by standard lifted inference techniques will also be reflected in the corresponding linear program. To verify whether this is indeed the case we constructed pairwise MRFs of varying size. We scaled the number of random variables from 25 to 625 arranged in a grid with pairwise and singleton factors with identical potentials. The results of the experiments can be seen in Figs. 4(a) and (b). As Fig. 4(a) shows, the number of LP variables is significantly reduced. Not only is the linear program reduced, but due to the fact that the lifting is carried out only once, we also measure a considerable decrease in running time as depicted in Fig. 4(b). Note that the time for the lifted experiment includes the time needed to compile the LP. This affirmatively answers **(Q1)**.

[1] the implementation can be found at [25].

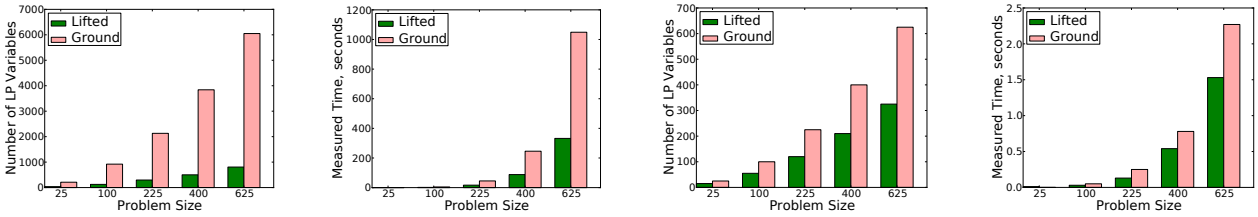
[2] `http://abel.ee.ucla.edu/cvxopt/`

**(Q2, Q3) Lifted MDPs:** Another application of linear programs that we considered is the computation of the value function in a Markov Decision Problem (MDP). The LP formulation of this task is as follows [22]:

$$\max_{\mathbf{v}} \ \mathbf{1}^T\mathbf{v}, \quad \text{s.t.} \ \ v_i \leq c_i^k + \gamma \sum\nolimits_{j\in\Omega_S} p_{ij}^k v_j,$$

where $v_i$ is the value of state $i$, $c_i^k$ is the reward that the agent receives when carrying out action $k$ and $p_{ij}^k$ is the probability of transfering from state $i$ to state $j$ by the action $k$. The MDP instance that we used is the well-known Gridworld (see e.g. [35]). The gridworld problem consists of an agent navigating within a grid of $n \times n$ states. Every state has an associated reward $R(s)$. Typically there is one or several states with high rewards, considered the goals, whereas the other states have zero or negative associated rewards.

At first we considered an instance of gridworld with a single goal state in the upper-right corner with a reward of 100. The reward of all other states was set to $-1$. As can be seen in Fig. 4(c), this example can be compiled to about half the original size. Fig. 4(d) shows that already this compression leads to improved running time. We now introduce additional symmetries by putting a goal in every corner of the grid. As one might expect this additional symmetry gives more room for compression, which further improves efficiency as reflected in Figs. 4(e) and 4(f). The two experiments presented so far affirmatively answer question **(Q1)**. However, the examples that we have considered so far are quite sparse in their structure. Thus, one might wonder whether the demonstrated benefit is achieved only because we are solving sparse problem in dense form. To address this we convert the MDP problem to a sparse representation for our further experiments. We scaled the number of states up to 1600 and as one can see in Fig. 4(g) and (h) lifting still results in an improvement of size as well as running time. Therefore, we can conclude that lifting an LP is beneficial regardless of whether the problem is sparse or dense, thus one might view symmetry as a dimension orthogonal to sparsity which answers question **(Q3)**. Furthermore, in Fig. 4(h) we break down the measured total time for solving the LP into the time spent on lifting and solving respectively. This presentation exposes the fact that the time for lifting dominates the overall computation time. Clearly, if lifting was carried out in every iteration (CVXOPT took on average around 10 iterations on these problems) the approach would not have been competitive to simply solving on the ground level. This justifies that the loss of potential lifting we had to accept in order to not carry out the lifting in every iteration indeed pays off **(Q2)**. Remarkably, these results follow closely what has been achieved

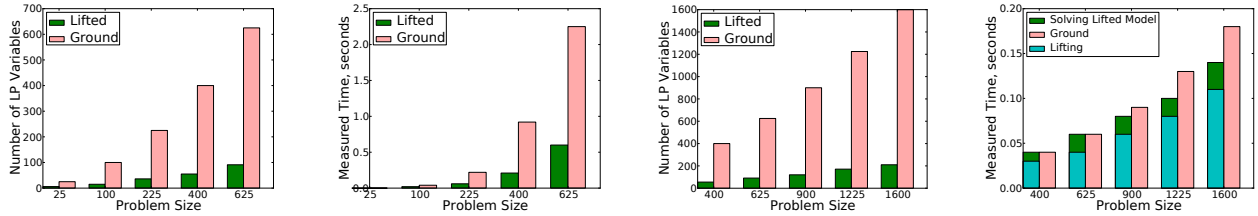



(a) Number of variables in the lifted and ground LPs.

(b) Time for solving the ground LP vs. time for lifting and solving.

(c) Ground vs. lifted variables on a basic gridworld MDP.

(d) Measured times on a basic gridworld MDP.




(e) Variables on a gridworld with additional symmetry.

(f) Measured times on a gridworld with additional symmetry.

(g) Variables on a gridworld with additional symmetry in sparse form.

(h) Measured times on a gridworld with additional symmetry in sparse form.

Figure 4: Experimental results (best viewed in color).

with MDP-specific symmetry-finding and model minimization approaches [26, 29, 13].

## 7 Discussion and Conclusion

We presented the first application of lifted inference techniques to linear programming. The resulting lifted linear programming approach compiles a given LP into an equivalent but potentially much smaller LP by grouping variables respectively constraints that are indistiguishable given the objective function and apply a standard LP solver to this lifted LP. The experimental results show that efficiency gains can be achieved.

Indeed, the link established here is related to symmetry-aware approaches in (mixed–)integer programming [23]. However, they are vastly different to LPs in nature. Symmetries in ILP are used for pruning the symmetric branches of search trees, thus the dominant paradigm is to add symmetry breaking inequalities, similarly to what has been done for SAT and CSP [31]. In contrast, lifted linear programming achieves speed-up by reducing the problem size. Furthermore, state-of-the-art symmetry detection for ILPs computes so-called orbit partition of the graph whose colored adjacency matrix is the skeleton equation. This is a "graph isomorphism"-complete problem, whereas our approach detects symmetries in time $\mathcal{O}(n^2 \log n)$ [6]. Moreover, the orbit partition of a graph is a refinement of the coarsest equitable partition, thus our approach results in more compression.

Regarding LPs, the work by Boedi *et al.* is probably the closest in spirit [5]. They showed that the set of combinatorial symmetries of the polytope that respect the objective can be used for compression. However, no polynomial algorithm for finding those symmetries was presented; instead they fell back to orbit partition-based methods in their experiments.

Given the current surge of interest in lifted inference, probably the most promising avenue for future work is to establish a similar strong link between MAP inference and LPs at the lifted level as it is known for the ground level [38, 20, 15, 21, 34]. Since, random variables easily become correlated within complex applications by virtue of sharing propagated evidence, one should develop approximate lifted LP approaches to still gain compression. Exploring the close connection to symmetry breaking in ILPs, CSPs, and MDPs and how the ideas carry over to lifted LPs is a promising future direction. Given the success of relational languages for probabilistic models, one should develop a relational LP specification language and exploit it for lifted linear programming. As lifting LPs itself is a major advance, its application to other AI and machine learning tasks and techniques such as regression and experimental design is another important direction.

**Acknowledgments:** The authors thank the anonymous reviewers for their valuable comments. This work was partly supported by the Fraunhofer ATTRACT fellowship STREAM and by the EC under contract number FP7-248258-First-MM.

## References


[1] B. Ahmadi, K. Kersting, and S. Sanner. Multi-Evidence Lifted Message Passing, with Application to PageRank and the Kalman Filter. In *Proc. of the 22nd International Joint Conference on Artificial Intelligence (IJCAI–11)*, Barcelona, Spain, July 16–22 2011.

[2] C.J. Augeri. *On Graph Isomorphism and the Pagerank Algorithm*. PhD thesis, Air Force Institute of Technology, WPAFB, Ohio, USA, 2008.

[3] D. Bickson, O. Shental, and D. Dolev. Distributed Kalman filter via Gaussian belief propagation. In *Proc. of the 46th Annual Allerton Conference on Communication, Control and Computing*, pages 628–635, Sept. 2008.

[4] D. Bickson, Y. Tock, O. Shental, and D. Dolev. Polynomial linear programming with Gaussian belief propagation. In *Proc. of the 46th Annual Allerton Conference on Communication, Control and Computing*, pages 895–901, Sept. 2008.

[5] R. Bödi, K. Herr, and M. Joswig. Algorithms for highly symmetric linear and integer programs. *Mathematical Programming, Series A*, Online First, Jan. 2011.

[6] P. Boldi, V. Lonati, M. Santini, and S. Vigna. Graph fibrations, graph isomorphism, and PageRank. *RAIRO , Informatique Théorique*, 40:227–253, 2006.

[7] D.M. Chickering and D. Heckerman. Targeted advertising with inventory management. In *In Proc. of the 2nd ACM Conference on Electronic Commerce (EC-00)*, pages 145–149, 2000.

[8] J. Choi and E. Amir. Lifted inference for relational continuous models. In *Proc. of the 26th Conference on Uncertainty in Artificial Intelligence (UAI-10)*, 2010.

[9] G. Dantzig and M. Thapa. *Linear Programming 2: Theory and Extensions*. Springer, 2003.

[10] L. De Raedt. *Logical and Relational Learning*. Springer, 2008.

[11] L. De Raedt, P. Frasconi, K. Kersting, and S.H. Muggleton, editors. *Probabilistic Inductive Logic Programming*, volume 4911 of *Lecture Notes in Computer Science*. Springer, 2008.

[12] R. de Salvo Braz, E. Amir, and D. Roth. Lifted First Order Probabilistic Inference. In *Proc. of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05)*, pages 1319–1325, 2005.

[13] T. Dean and Robert Givan. Model minimization in markov decision processes. In *Proc. of the Fourteenth National Conference on Artificial Intelligence (AAAI-97)*, pages 106–111, 1997.

[14] L. Getoor and B. Taskar, editors. *An Introduction to Statistical Relational Learning*. MIT Press, 2007.

[15] A. Globerson and T. Jaakkola. Fixing max-product: Convergent message passing algorithms for map LP-relaxations. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems (NIPS)*, 2007.

[16] V. Gogate and P. Domingos. Probabilistic theorem proving. In *Proc. ot the 27th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2011.

[17] W.H. Haemers. Interlacing eigenvalues and graphs. *Linear Algebra and its Applications*, 226/228:593–616, 1995.

[18] R.A. Horn and C.A. Johnson, editors. *Matrix Analysis*. Cambridge University Press, 1985.

[19] K. Kersting, B. Ahmadi, and S. Natarajan. Counting Belief Propagation. In *Proc. of the 25th Conference on Uncertainty in Artificial Intelligence (UAI–09)*, 2009.

[20] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(10):1568–1583, 2006.

[21] V. Kolmogorov and C. Rother. Minimizing nonsubmodular functions with graph cuts-a review. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(7):1274–1279, 2007.

[22] M.L. Littman, T.L. Dean, and L. Pack Kaelbling. On the complexity of solving markov decision problems. In *Proc. of the 11th International Conference on Uncertainty in Artificial Intelligence (UAI-95)*, pages 394–402, 1995.

[23] F. Margot. Symmetry in integer linear programming. In M. Jünger, T.M. Liebling, D. Naddef, G.L. Nemhauser, W.R. Pulleyblank, G. Reinelt, G. Rinaldi, and L.A. Wolsey, editors, *50 Years of Integer Programming 1958–2008: From the Early Years to the State-of-the-Art*, pages 1–40. Springer, 2010.

[24] B. Milch, L. Zettlemoyer, K. Kersting, M. Haimes, and L. Pack Kaelbling. Lifted Probabilistic Inference with Counting Formulas. In *Proc. of the 23rd AAAI Conf. on Artificial Intelligence (AAAI-08)*, July 13-17 2008.

[25] M. Mladenov, B. Ahmadi, and K. Kersting. An implementation of lifted linear programming. http://www-kd.iai.uni-bonn.de/index.php?page=software_details&id=25, 2012.

[26] S.M. Narayanamurthy and B. Ravindran. On the hardness of finding symmetries in markov decision processes. In *Proc. of the 25th International Conference on Machine Learning (ICML-08)*, pages 688–695, 2008.

[27] D. Poole. First-Order Probabilistic Inference. In *Proc. of the 18th International Joint Conference on Artificial Intelligence (IJCAI-05)*, pages 985–991, 2003.

[28] F. Potra and S. J. Wright. Interior-point methods. *Journal of Computational and Applied Mathematics*, 124:281–302, 2000.

[29] B. Ravindran and A.G. Barto. Symmetries and model minimization in markov decision processes. Technical Report 01-43, University of Massachusetts, Amherst, MA, USA, 2001.

[30] M. Richardson and P. Domingos. Markov Logic Networks. *Machine Learning*, 62:107–136, 2006.

[31] M. Sellmann and P. Van Hentenryck. Structural symmetry breaking. In *in Proc. of 19th International Joint Conference on Artificial Intelligence (IJCAI-05)*, 2005.

[32] P. Sen, A. Deshpande, and L. Getoor. Exploiting Shared Correlations in Probabilistic Databases. In *Proc. of the Intern. Conf. on Very Large Data Bases (VLDB-08)*, 2008.

[33] P. Singla and P. Domingos. Lifted First-Order Belief Propagation. In *Proc. of the 23rd AAAI Conf. on Artificial Intelligence (AAAI-08)*, pages 1094–1099, Chicago, IL, USA, July 13-17 2008.

[34] D. Sontag, A. Globerson, and T. Jaakkola. Clusters and coarse partitions in LP relaxations. In *Proceedings of the 22nd Annual Conference on Neural Information Processing Systems (NIPS)*, pages 1537–1544, 2008.

[35] R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998.

[36] G. Van den Broeck, N. Taghipour, W. Meert, J. Davis, and L. De Raedt. Lifted probabilistic inference by first-order knowledge compilation. In *Proc. of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2178–2185, 2011.

[37] Y. Weiss and W.T. Freeman. Correctness of belief propagation in gaussian graphical models of arbitrary topology. *Neural Computation*, 13(10):2173–330, 2001.

[38] C. Yanover, T. Meltzer, and Y. Weiss. Linear programming relaxations and belief propagation an empirical study. *Journal of Machine Learning Research*, 7:1887–1907, 2006.