2011 23rd IEEE International Conference on Tools with Artificial Intelligence

Independence-based MAP for Markov networks structure discovery

Facundo Bromberg, Federico Schlüter, Alejandro Edera
Departamento de Sistemas de Información,
Universidad Tecnológica Nacional, Facultad Regional Mendoza, Argentina.
{fbromberg,federico.schluter,aedera}@frm.utn.edu.ar

***Abstract*—This work presents IBMAP, an approach for robust learning of Markov network structures from data; together with IBMAP-HC, an efficient instantiation of the approach. Existing Score-Based (SB) and Independence-Based (IB) approaches must make concessions either on robustness or efficiency. IBMAP-HC improves robustness efficiently through an IB-SB hybrid approach based on the probabilistic Maximum-A-Posteriori (MAP) technique; and the IB-score, a tractable expression for computing posterior probabilities of Markov network structures. Performance is first tested against IB and SB competitors on synthetic datasets. Against IB competitors (GSMN algorithm and a version of the HHC algorithm adapted here for Markov networks discovery), IBMAP-HC showed reductions in edges Hamming distance with same order running times. Against SB competitors, both IBMAP-HC and our adaptation of HHC produced comparable Hamming distances, but with running times orders of magnitude faster. We also evaluated IBMAP-HC in a realistic, challenging testbed: EDAs, evolutionary algorithms for optimization that estimate a distribution on each generation. Using IBMAP-HC to estimate distributions, EDAs converged to the optimum faster in all benchmark functions considered, reducing required fitness evaluations by up to 80%.**

***Keywords*-Graphical Models; Markov networks; Structure Learning; Estimation of Distribution algorithms;**

I. Introduction

We present in this work the **IBMAP** (***I**ndependence-**B**ased **M**aximum **a** **P**osteriori*) approach for robust learning of Markov network (**MN**) structures from data (1; 2); together with **IBMAP-HC** (for hill-climbing), an efficient instantiation of the approach. MNs, together with *Bayesian networks* (**BN**) belong to the family of *probabilistic graphical models*, a computational framework for compact representation of joint probability distributions. They consist of an undirected (MNs) or directed (BNs) graph G and a set of numerical parameters $\boldsymbol{\theta}$. Each of the n nodes in G represents a random variable, and the edges (undirected or directed) encode conditional independences between them. The graph is therefore commonly called the *independence structure* of the distribution, or simply, the *structure*. The importance of these independencies is that they factorize the joint distribution over the whole set of variables into factors over subsets of variables, resulting in important reductions in the dimensionality of the distribution.

An interesting problem is learning these models from data, consisting in learning both G and $\boldsymbol{\theta}$. In many practical scenarios the goal of knowledge discovery, or the problem of understanding a domain, is clearly distinguished from the goal of inference, or the problem of predicting variable values under different contexts. In this work we concentrate on the sub-problem of learning the structure of interactions among variables, an important source of information in knowledge discovery. This problem has resulted in important contributions during recent years, although many of its core difficulties remain a challenge and are under intense work.

The literature shows two main approaches for learning structures from data: ***I**ndependence-**B**ased* (**IB**) and *Score-**B**ased* (**SB**) algorithms. IB algorithms proceed by first learning G, and then, given G, estimate $\boldsymbol{\theta}$. To learn G they perform a succession of *statistical independence tests* (3), or simply *tests* hereon. Each test responds to independence queries between two variables X and Y in the domain of variables $\mathbf{V}$, given some subset $\mathbf{Z}$ of variables in $\mathbf{V}$. We denote the independence and dependence between this triplet $\langle X, Y, \mathbf{Z}\rangle$ by $(X \perp\!\!\!\perp Y | \mathbf{Z})$ and $(X \not\perp\!\!\!\perp Y | \mathbf{Z})$, respectively. Examples of tests used in practice are Mutual Information, Pearson's χ^2, G^2 (4), the *Bayesian* test (5; 6), and for continuous Gaussian data the *partial correlation* test (3), among others. For learning G, IB algorithms proceed iteratively. On each iteration they decide first the test to be executed based on the independences learned so far, and then, discard all those structures inconsistent with the independence outcome of the test; until a single structure is left.

For BNs, the IB approach has been mainly exemplified by the SGS and PC algorithms (3), and the family of structure learning algorithms generalized by the local-to-global (LGL) framework of (7). LGL is a framework for producing structure learning algorithms from algorithms that learn the local neighborhood of variables (according to the Generalized Local Learning framework of (8)). The Grow Shrink algorithm, and many improvements widely used in the field such as IAMB and its variants, HITON-MB, and MMMB learn the probabilistic local neighborhood called the *Markov blanket* (defined formally in Section II-A1). The Grow Shrink algorithm has been used for learning BNs (9), but for BNs, it is possible to learn the structure from smaller neighborhoods formed by the parent (P) and children (C) of a node in the network, resulting in the HITON-PC and MMPC local algorithms, and the corresponding structure learning algorithms HHC and MMHC, respectively. Adaptations of the LGL approach based on the Grow Shrink local learning algorithm has been proposed for MNs, resulting in the GSMN algorithm (10). In this work we consider HHC as a competitor for IBMAP-HC as it is one of the best IB algorithms for BNs according to results presented in (7). We thus present in the experimental section an adaptation of HHC for learning MNs, denoted here HHC-MN.

The learning of numerical parameters $\boldsymbol{\theta}$ for MNs is an NP-hard problem (11), solved in practice through a data-intensive numerical algorithm (12). IB algorithms learn G without estimating $\boldsymbol{\theta}$, contrary to SB algorithms as explained below. They are therefore efficient, reaching in some cases polynomial complexities in the number of tests; e.g., the GSMN algorithm requires $O(n^2)$ tests. Another advantage of these algorithms is that they are amenable to proof of correctness under the assumptions of *faithfulness* of the graph to the model (i.e., the graph can encode the true underlying independences), *positivity* of the underlying model, and *tests reliability*. To the best of the authors knowledge, the first assumption is made by all existing IB and SB algorithms, whereas the second assumption is made by all existing IB algorithms. The third assumption is violated unless the datasets are sufficiently large. Both approaches benefit then from larger datasets, but the problem is exacerbated for IB algorithms as tests reliability degrades exponentially with the amount of variables involved (for some fixed size dataset). For good quality, these tests require enough counts in their contingency tables, and there are exponentially many of those (one per value assignment of all variables in the tests). For example, for the χ^2 test (13) recommends that the test be deemed unreliable if more than 20% of these cells have an expected count of less than 5 data points.

SB algorithms (14; 15; 16), approach the problem of structure discovery as an optimization problem on the space of structures and the space of parameters, looking for the one with maximum *score*. Examples of scores are maximum likelihood, minimum description length (17), and Pseudo-likelihood (18; 15), among others. An advantage of these algorithms is their resiliency to data scarceness, at the expense of important computational costs when learning MNs. For each structure during the search, they require the estimation of $\boldsymbol{\theta}$; computationally intensive for requiring an inference step. Also, to avoid over-fitting, many of these methods consider a regularization term adding an extra hyper-parameter λ, whose best value has to be found empirically (e.g., running the training stage for several λ's, potentially with cross-validation).

The rest of this work is organized as follows. Section II presents IBMAP, an IB-SB hybrid approach. This approach is based on a MAP search of the structure with maximum IB-score, which is a score for structures that can be computed efficiently with $O(n^2)$ tests. This section also introduces an efficient instance of IBMAP, called IBMAP-HC. Detailed and systematic comparisons between IBMAP-HC and representatives of the IB and SB approaches are presented in the experimental Section III. The results show that our solution improves significantly the quality of current state-of-the-art IB approaches, with comparable runtimes; and results in outputs with quality comparable to state-of-the-art SB approaches, with orders of magnitude faster runtimes. Section IV further evaluates the performance of IBMAP-HC against competitors in a more realistic scenario: *Estimation of Distribution algorithms* (**EDAs**). Finally, the paper concludes with a summary and possible directions of future work in Section V.

II. THE IBMAP APPROACH

We introduce now our IBMAP approach for learning the structure of a MN from data (denoted $\mathcal{D}$ hereon). IBMAP applies the well known model selection MAP statistical technique to the structure selection problem:

$$G^\star = \arg\max_G \Pr(G \mid \mathcal{D}). \qquad (1)$$

IBMAP is a hybrid of the IB and SB approaches. It is SB in that it follows a MAP approach, with the score being $\Pr(G \mid \mathcal{D})$. It is IB in that it learns G by performing tests on data. The main difference of IBMAP with existing SB algorithms is that $\Pr(G \mid \mathcal{D})$ as a score ranks structures G, not models $(G, \boldsymbol{\theta})$.

As mentioned, tests require exponentially large datasets to respond with total certainty about conditional independencies, leading in practice to erroneous independence assertions. Most of the existing IB algorithms simply ignore this fact and proceed, potentially discarding the true structure. Like (6), we model this uncertainty by the posterior probabilities $\Pr(c|\mathcal{D})$ of independence assertions c, and relate them to the posterior of the structure G through the concept of *closure*, denoted $\mathcal{C}(G)$, and defined as the set of independence assertions c_i, $i = 1, \ldots, |\mathcal{C}(G)|$, that completely determine the structure G. The posterior of G can be therefore equated to the posterior of the closure, i.e.,

$$\Pr(G \mid \mathcal{D}) = \Pr(\mathcal{C}(G) \mid \mathcal{D}).$$

To compute the posterior $\Pr(\mathcal{C}(G) \mid \mathcal{D})$ we can apply the chain rule over all its independence assertions obtaining $\prod_{i=1}^{|\mathcal{C}(G)|} \Pr(c_i|c_1, \ldots, c_{i-1}, \mathcal{D})$. To the best of the authors knowledge, there is no existing method for computing probabilities of independence assertions conditioned on other independence assertions and data. There is, however, the Bayesian test for computing the posteriors of independence assertions (5). To apply this test we should eliminate independence assertions from the conditioning, that is, assume independence assertions are mutually independent given $\mathcal{D}$. Interestingly, this assumption is made by all IB algorithms that, based on the independencies learned so far, decide on which independence to test next, but not the value of these independencies, which are only influenced by the input data

$\mathcal{D}$. The result of this approximation is the IB-score $\sigma(G)$ of G, as a factorization of the posterior of the closure in the following product of posteriors of independence assertions:

$$\sigma(G) = \Pr(\mathcal{C}(G) \mid \mathcal{D}) \approx \prod_{c \in \mathcal{C}(G)} \Pr(c \mid \mathcal{D}). \tag{2}$$

A. THE IBMAP-HC ALGORITHM

We describe now IBMAP-HC, our efficient instantiation of IBMAP, by instantiating the closure and the MAP search.

1) Markov Blanket Closure: We propose a closure based on *Markov blankets*. The Markov blanket of a variable $X \in \mathbf{V}$ is a set $\mathbf{B}_X \subseteq \mathbf{V} - \{X\}$ of variables that "shields" X of the probabilistic influence of variables not in $\mathbf{B}_X$, i.e., for every $X \neq Y \in \mathbf{V}$, and $Y \notin \mathbf{B}_X$, $(X \perp\!\!\!\perp Y | \mathbf{B}_X - \{Y\})$. It has been shown (2) that for Markov networks the Markov blanket $\mathbf{B}_X$ of variable X corresponds to the neighbors of X in the network.

The proposed Markov-blanket closure is:

$$\mathcal{C}(G) = \bigcup_{X \in \mathbf{V}} \mathcal{C}_X(G), \tag{3}$$

where each $\mathcal{C}_X(G)$ is the union of two mutually exclusive sets:

$$\begin{aligned}\mathcal{C}_X(G) = &\Big\{(X \not\perp\!\!\!\perp W | \mathbf{B}_X \backslash W) \ : \ W \in \mathbf{B}_X\Big\} \cup \\ &\Big\{(X \perp\!\!\!\perp W | \mathbf{B}_X \backslash W) \ : \ W \notin \mathbf{B}_X\Big\},\end{aligned} \tag{4}$$

that is, for each $W \in \mathbf{B}_X$ add $(X \not\perp\!\!\!\perp W | \mathbf{B}_X \backslash W)$, or add $(X \perp\!\!\!\perp W | \mathbf{B}_X \backslash W)$ otherwise.

The decomposition of the closure into closures for each variable results in a decomposition of the score into a product of *variable* scores: $\sigma(G) = \prod_{X \in \mathbf{V}} \sigma_X(G)$. This decomposition allows for an *incremental* computation of the score $\sigma(G')$ based on the score $\sigma(G)$ of a structure G differing with G' by a number k of edges. We start considering the case that G and G' differ by a single edge (X, Y). Being the blankets the set of neighbors in the structure, the edge difference results in only blankets $\mathbf{B}_X$ and $\mathbf{B}_Y$ differing between the structures, and therefore only score σ_X and σ_Y has to be recomputed to obtain the score of G', i.e., $\sigma(G') = \sigma(G)\ [\sigma_X(G')\sigma_Y(G')]\ /\ [\sigma_X(G)\sigma_Y(G)]$, with a cost of $O(n)$ tests. For k edges differences between the structures, at most $2k$ blankets are affected, and therefore at most $2k$ factors has to be re-computed, again a cost of $O(n)$ tests.

The incremental computation of the score may have an important impact in local search optimization algorithms, such as the IBMAP-HC algorithm described in the next section, that proceed by exploring successively structures a few edges apart.

At this point, it still remains to prove that $\mathcal{C}(G)$ as defined by Eq. (3) is in fact a closure, i.e., that its independence assertions determine the structure. For that let us first present the following theorem:

Theorem 1 (Pearl and Paz 1987 (19)): Every positive probability distribution P on $\mathbf{V}$, has a unique Markov network G satisfying, for every $X \neq Y \in \mathbf{V}$,

$$(X \perp\!\!\!\perp Y | \mathbf{B}_X \backslash Y) \text{ in } P \ \Leftrightarrow \ (X \perp\!\!\!\perp Y | \mathbf{V} \backslash \{X, Y\}) \text{ in } P \tag{5}$$

$$(X, Y) \notin E(G) \ \Leftrightarrow \ (X \perp\!\!\!\perp Y | \mathbf{V} \backslash \{X, Y\}) \text{ in } P \tag{6}$$

where $E(G)$ denotes the set of edges in structure G.

To prove that the closure determines the no-edge between X and Y in G, we first note that a no-edge implies $Y \notin \mathbf{B}_X$, and therefore the closure should contain exactly the independence assertion of the l.h.s. of Eq.(5), which by transitivity with the equivalence of Eq.(6), determines the no-edge (the same result is obtained starting from the fact that $X \notin \mathbf{B}_Y$). Edges in G are similarly determined using the counter-positive of these equivalences.

2) Hill-climbing MAP Search : To conclude the presentation of our approach we present IBMAP-HC, an efficient implementation of the MAP search of the structure with maximum IB score σ, computed over the independence assertions given by the Markov blanket closure $\mathcal{C}$.

IBMAP-HC is outlined in Algorithm 1. It performs a hill-climbing search in the space of structures, *starting* in the empty structure G_i, $i = 0$ (line 2), *selecting* in line 4 a candidate $\tilde{G}_{i+1}$, and *terminating* in line 6 if such candidate does not improve the exact IB score $\sigma(G_i)$ of G_i. To select the candidate, instead of using the neighbor with maximum IB score σ, it selects $\tilde{G}_{i+1}$, the neighbor that maximizes an approximation of the IB score, denoted $\tilde{\sigma}$. As the set of neighbors of G_i, we considered the set of structures one edge-*flip* away from G_i, denoted $\mathcal{N}^1(G_i)$, where a *flip* consists in removing an edge if such edge exists, or adding it otherwise.

Algorithm 1 IBMAP-HC
```
1: $i \leftarrow 0$
2: $G_0 \leftarrow$ empty structure;
3: **repeat** over increasing values of integer $i$
4:         $\tilde{G}_{i+1} \leftarrow \arg\max_{G' \in \mathcal{N}^1(G_i)} \tilde{\sigma}(G')$
5: **until** $\sigma(\tilde{G}_{i+1}) < \sigma(G_i)$
6: **return** $G_i$
```

By selecting a potentially sub-optimal neighbor, the approximation may result in the algorithm terminating before reaching a local maxima. However, this comes along with important computational savings, from $O(n^4)$ to $O(n)$ tests per iteration, and an overall final cost of $O(n^2)$ tests; as we proceed to explain. If the maximization of line 4 were performed over the exact IB score, the cost of the maximization would be $O(n^4)$ tests; that is, $\binom{n}{2}$ IB score calculations of $O(n^2)$, one for each structure in $\mathcal{N}^1$. The

Table I: HDs, runtime, λ^* (only SB) for GSMN, HHC-MN, IBMAP-HC, and SB algorithms.

		n=20								
		HD				**RUNTIME** (ms)				$\boldsymbol{\lambda^*}$
τ	D	**GSMN**	**HHC-MN**	**IBMAP-HC**	**SB**	**GSMN**	**HHC-MN**	**IBMAP-HC**	**SB**	**SB**
2	25	42	30	17	20	242	41	67	949	0.300
	50	49	16	18	17	84	36	34	2472	0.150
	100	50	16	15	14	111	18	40	21778	0.080
4	25	49	33	41	35	203	60	40	3718	0.240
	50	45	32	31	32	89	64	66	12400	0.150
	100	45	22	30	34	101	95	69	15104	0.160
8	25	79	83	78	76	15	144	76	772561	0.040 6
	50	83	80	85	71	45	124	74	990802	0.010
	100	70	76	72	72	57	232	85	1227105	0.009

first thing to note is that all neighbors $G' \in \mathcal{N}^1(G_i)$ differ by only one edge with G_i. Also, at each iteration of the loop the score of G_i was already computed in the previous iteration, we can therefore compute the score of G' incrementally as explained in the previous section, resulting in a cost of the maximization of $\binom{n}{2}$ computations of $O(n)$ each. The algorithm computes instead the approximate score $\widetilde{\sigma}(G')$, for some $G' \in \mathcal{N}^1(G_i)$. As discussed in the previous section, when differing by an edge (X, Y), assertions in $\mathcal{C}_X$ and $\mathcal{C}_Y$ of the two structures differ as well. The score of G' is computed approximately assuming that the posteriors of all these assertions are not affected, with the exception of assertions corresponding only to triplets over both X and Y, i.e., $\langle X, Y, \mathbf{B}_X \setminus Y\rangle$ and $\langle Y, X, \mathbf{B}_Y \setminus X\rangle$. These assertions have complementary values in G' and G_i. For instance, if $(X, Y) \in E(G_i)$ and $(X, Y) \notin E(G')$, then Eq. (3) indicates that $(X \not\perp\!\!\!\perp Y | \mathbf{B}_X \setminus Y) \in \mathcal{C}_X(G_i)$ while $(X \perp\!\!\!\perp Y | \mathbf{B}_X \setminus Y) \in \mathcal{C}_X(G')$, and $(Y \not\perp\!\!\!\perp X | \mathbf{B}_Y \setminus X) \in \mathcal{C}_Y(G_i)$ while $(Y \perp\!\!\!\perp X | \mathbf{B}_Y \setminus X) \in \mathcal{C}_Y(G')$. Being complementary, i.e., they add up to 1, the posteriors for G' can be computed from those of G_i, which were already computed in the previous iteration of the main loop. We can compute the posteriors for G', and the whole approximate score $\widetilde{\sigma}(G')$ in constant time, resulting in a reduction from $O(n^4)$ tests in the computation of the maximization of line 4, to no test computation at all.

Although apparently rather drastic, the approximation is well justified by noticing that all ignored independence assertions maintain their truth value and differ only in the conditioning sets having one variable more or one variable less (depending on whether which structure has the extra edge). For instance, if G' has the extra edge (X, Y), the independence assertion $(X \perp\!\!\!\perp W | \mathbf{B}_X \setminus W)$ in $\mathcal{C}_X(G_i)$ becomes $(X \perp\!\!\!\perp W | \mathbf{B}_X, Y \setminus W)$ in $\mathcal{C}_X(G')$, for some arbitrary $W \in \mathbf{V} \setminus \{X, Y\}$. The approximation is reduced to assuming that two independence assertions differing by one variable in the conditioning set have similar posteriors.

At this point the most expensive operation in the main loop becomes the computation of the (exact) IB score of $\widetilde{G}_{i+1}$ at line 6, with a cost of $O(n)$ when computed incrementally; resulting in an overall computational cost for the loop of $O(M \cdot n)$, M denoting the number of iterations until terminate. To this cost, it only remains to add the cost of computing the initial structure G_0, with a cost of $O(n^2)$, resulting in an overall cost for the algorithm of $O(n^2 + Mn)$. Since M can be obtained only empirically, experiments were performed to show that M is not a source of an extra degree in the complexity by showing it grows at most linearly with n. This result is shown in the next section.

III. EVALUATION OF STRUCTURE LEARNING

This section describes several experiments for testing empirically the effectiveness of IBMAP-HC in improving the quality of structures discovered by existing IB algorithms in equivalent runtimes, and obtaining competitive qualities with SB algorithms with lower runtimes.

As IB competitors we considered GSMN as a well established MN structure learning algorithm (10), and HHC, a state of the art, robust IB structure learning algorithm of BNs (8), adapted here for MNs. For a fair comparison against IBMAP-HC, we run both algorithms with the Bayesian test for statistical independence queries.

The GSMN algorithm is an efficient algorithm that computes only $O(n^2)$ tests, constructing the structure from the Markov blanket of each variable; learned with the *Grow-Shrink* algorithm (9). HHC instead, learns the structure by learning the set of parents and children (PC) of each variable through the interleaved HITON-PC with symmetry correction algorithm (20; 8). This is in fact possible for BNs, even though the Markov Blanket of a variable is composed not only by the PC set but also by the spouses of the variable (i.e., the other parents of its children). Interleaved HITON-PC executes at each iteration a step exponential in the size of the current estimate of the PC set. In many practical situations, however, a node may have many parents, and thus the PC of any of its parents may be much smaller than its Markov Blanket. For this reason the algorithm has shown in practice to scale to thousands of variables despite this exponentiality (see a thorough experimental evaluation in (8)). For the case of MNs, being undirected graphs, the equivalent of the PC of a variable is its neighbors, which is exactly its Markov Blanket. It is therefore expected that

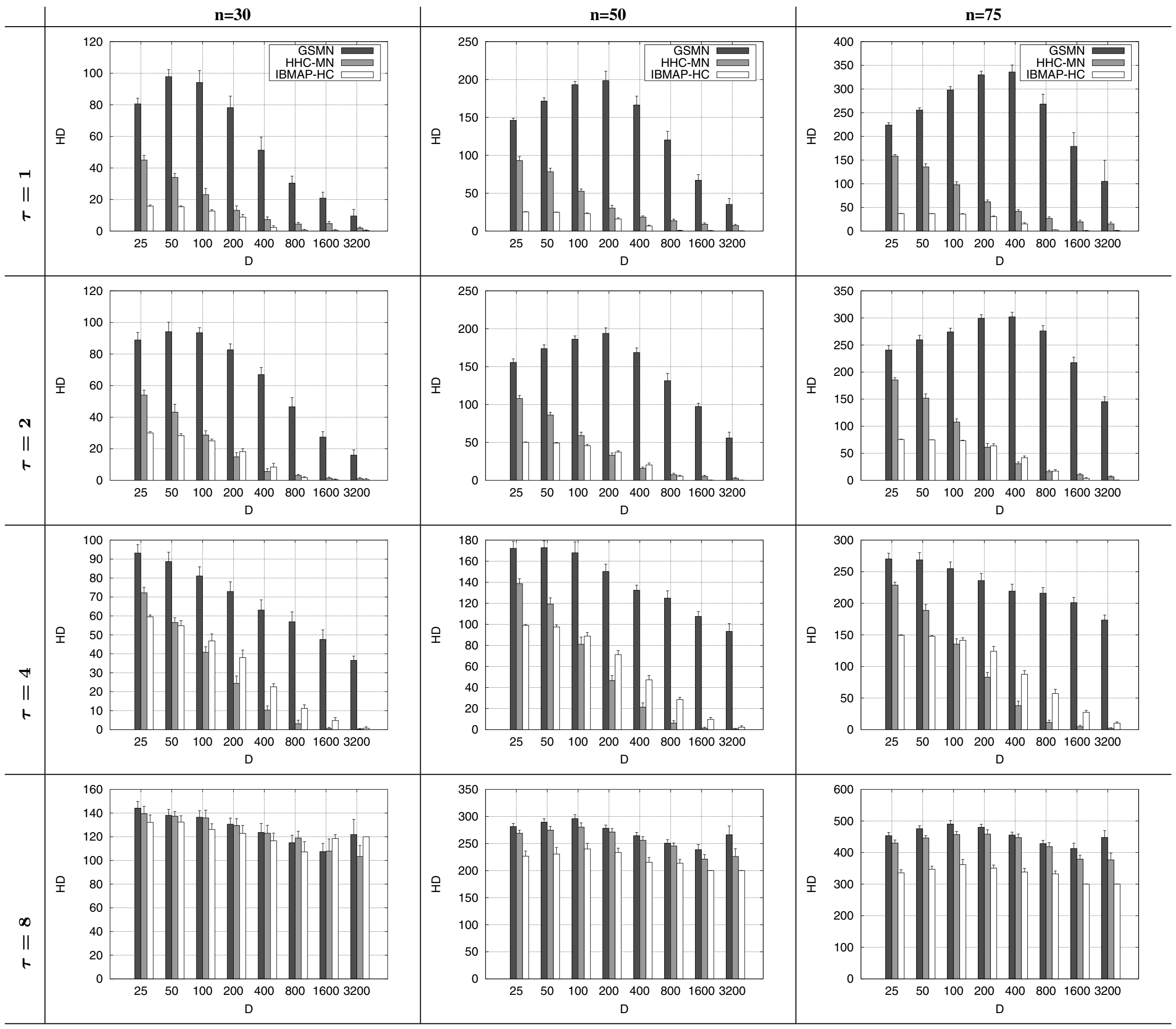

Figure 1: Average of HDs for GSMN, HHC-MN and IBMAP-HC algorithms

HITON-PC learns the Markov Blanket of a MN, and thus it can be used as part of HHC to learn the undirected structure. To get a MNs learning algorithm we then simply omit the final step of HHC that orients the edges, denoting the resulting algorithm by HHC-MN. As a final remark, we note that being the PC and Markov Blanket sets equivalent in MNs, the savings gained for BNs are non-existent and thus HHC-MN is expected to scale to fewer variables than its BN counterpart.

As a representative of the SB approach we considered a variation of an L_1-regularized maximum Pseudo-likelihood estimator. It is a double-loop learning process that consists in a *Grafting*-based greedy search of the space of structures (16; 1). The outer loop check first the termination condition, and then proposes a new structure greedily by the *Grafting* criteria (21). Then, in the inner loop, the parameters for this new structure are learned by finding those that maximize the L_1-regularized Pseudo-likelihood of the model using the the LBFGS algorithm (22).

The experiments were conducted on synthetic datasets generated using a Gibbs sampler by sampling from known random models. This allows a systematic and controlled study, providing a known underlying structure, and a better assessment of the quality by comparing the edges *Hamming distance* (**HD**) between the learned structure and the underlying one, i.e., the sum of false positive and false negative edges of the learned structure. We report also the execution runtimes of the algorithms. The synthetic random models were generated for domains of $n = 20, 30, 50, 75$ binary variables. For each domain size, 10 random networks were generated for increasing connectivities $\tau = 1, 2, 4, 8$ by considering as edges the first $n\tau/2$ variable pairs of a random permutation of the set of all variable pairs. We considered here a pair-wise factoring of the models using only factors for cliques of size 2 (i.e., one factor per edge). The 4 parameters of each clique (one per configuration of the two binary variables) were generated randomly but assuring a strong enough dependence by forcing the log-odds of the two variables in the edge to be larger than 1.0. Given these models, datasets with increasing number of data-points $\mathcal{D} = 25, 50, 100, 200, 400, 800, 1600, 3200$ were sampled for each (n, τ) combination. Because of the regularization term, the SB algorithm was run for a sufficiently large range of λ values to assure they contain the model with lowest HD. For $\tau = 8$ (both n's) 10 values in the interval $[0.007, 0.1]$ were considered, whereas for $\tau = 2$ and $\tau = 4$, 10 values in the interval $[0.1, 0.5]$ were considered.

Due to the large runtimes of our SB implementation, comparisons against it were conducted for one repetition only, domains of $n = 20$ variables, connectivities $\tau = 2, 4, 8$ and datasets with $\mathcal{D} = 25, 50, 100$ datapoints. Resulting HDs and runtimes (in milliseconds) are shown in Table I. The HD reported for SB is the smallest found over the range of λ's. The corresponding λ, denoted λ^*, is shown in the neighboring column. Similarly, the runtime reported for SB corresponds to the single run of $\lambda = \lambda^*$. In practice, λ^* cannot be selected by this way as the underlying model is unknown (i.e., HD cannot be computed). This way, however, we abstract our results from any particular hyper-parameter λ selection procedure. Other methods of selecting λ^* can only result in equal or worst HD and runtime for SB, only improving the merits of IBMAP-HC and HHC-MN against it. The table shows that both, IBMAP-HC and HHC-MN qualities are better (lower HD) than that of GSMN in all cases except of $\tau = 8$, $\mathcal{D} = 100$, where GSMN shows HD reduction of up to 35% against IBMAP-HC. In contrast, IBMAP-HC and HHC-MN improve in all other cases, with comparable runtimes, as the corresponding columns show.

When compared to the SB algorithm, IBMAP-HC and HHC-MN also perform well, showing comparable HDs. As expected, both algorithms run much faster than the SB algorithm, with runtimes $1-4$ orders of magnitude faster in all cases. This difference in runtime may be up to 2 orders of magnitude larger in practice, as the SB algorithm must be ran for the whole range of λ's for finding λ^*; in some cases even requiring a cross-validation run per λ.

In a second experiment we conducted the comparison only among IB algorithms, for datasets with $n = 30, 50, 75$, $\tau = 1, 2, 4, 8$ and a more detailed increasing set of values $\mathcal{D} = 25, 50, 100, 200, 400, 800, 1600, 3200$. Figure 1 shows such comparison reporting the mean values and standard deviations of the HD of GSMN, HHC-MN and IBMAP-HC, for 10 different datasets. The graphs are ordered in the Figure by columns for different n values, and by rows for different τ values. The figure shows clearly that both, IBMAP-HC and HHC-MN qualities are better (lower HD)

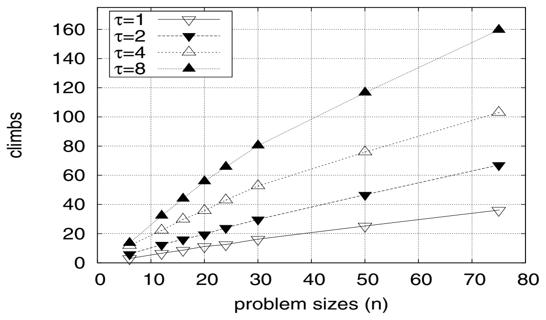

Figure 2: Number of IBMAP-HC climbs for $D = 1000$

than that of GSMN in all cases. IBMAP-HC presents better or equal (up to statistical significance) HDs than that of HHC-MN in all cases of $\tau = 1$; all cases of $\tau = 2$ except $\mathcal{D} = 200$ and 400; only for $\mathcal{D} = 25, 50$ in $\tau = 4$; and all cases of $\tau = 8$ except the two results of $\mathcal{D} = 1600, 3200$ of $n = 30$. The best improvements are obtained for $\tau = 1$ with reductions of IBMAP-HC's HD up to a 25% of HHC-MN's HD for $n = 75$ and low $\mathcal{D}$.

We do not show runtimes for $n = 30, 50, 75$, but similarly to $n = 20$, for all three algorithms the runtimes were similar in all τ's and $\mathcal{D}$'s. Instead, we show empirically the $O(n^2)$ runtime complexity of IBMAP-HC by showing that the number of climbs M of IBMAP-HC grows linearly with n (c.f. Section II-A2). Fig. 2 shows measurements of M for problems with increasing n, and $\tau = 1, 2, 4, 8$, $\mathcal{D} = 1000$, indicating that M grows linearly or slower.

IV. IBMAP-HC FOR EDAs

We present now results of evaluating IBMAP-HC in a practical scenario, the *Estimation of Distribution algorithms* (**EDA**). These are variations of the well-known evolutionary algorithms, that perform the same *selection* and *variation* stages, but replace the *crossover* and *mutation* stages for generating a new population, with the *estimation* and *sampling* stages. The former stage *estimate* a probability distribution from the current population, generating the next population by *sampling* from it (thus their name). In the *estimation* stage, EDAs estimate the probability distribution from the dataset corresponding to the current population. This is because they associate each gene to a random variable, each individual to a joint assignment of these variables, and the selected population to a sample of the distribution. The rationale for replacing selection with estimation is that by estimating the distribution from the selected individuals, that is, those best fitted, the sampling stage would produce novel, yet well-fitted individuals. For more details see (23; 24).

Several MN-*based* EDAs have been proposed recently that uses MNs for modeling the distribution, as (25; 26), among others. As a test-bed we considered the work of Shakya and Santana: the *Markovianity Optimization Algorithm* (**MOA**). This is an MN-based EDA that learns the MN

Table II: Results for Royal Road (left) and OneMax (right) problems

	Royal Road					OneMax			
	MOA		MOA'			MOA		MOA'	
n	D^*	f^*	D^*	f^*	n	D^*	f^*	D^*	f^*
16	100	545.00 (59.86)	50	337.50 (176.09)	15	50	267.50 (35.45)	50	202.50 (14.19)
32	400	3800.00 (210.82)	400	2140.00 (134.99)	30	200	1170.00 (94.87)	100	475.00 (42.49)
64	800	9120.00 (252.98)	800	4440.00 (126.49)	60	800	5200.00 (98.46)	200	1050.00 (52.70)
92	1600	18400.00 (533.33)	800	5080.00 (500.67)	90	800	5560.00 (126.49)	400	2220.00 (63.25)
120	1600	31120.00 (822.31)	1600	9840.00 (386.44)	120	1600	11200.00 (871.53)	800	4400.00 (312.33)

structure from the population using an efficient IB structure learning algorithm based on mutual information (MI), a simple IB structure learning algorithm described in detail in the same work. Sampling in MOA is conducted through a variation of a Gibbs sampler that requires only the structure of the model, avoiding the need to learn the model parameters. The implementation of MI in MOA takes advantage of experts information indicating the maximum number of neighbor variables that a variable can have, denoted here k. We tested MI and IBMAP-HC for different values of k (results not shown here), observing great sensitivity of MI to its value, while IBMAP-HC showed no loss of quality. In the experiments below we set the value of k for MI to be the closest to the true value, resulting in the best possible performance of MI, i.e., the strongest competitor for IBMAP-HC.

We conducted experiments to compare IBMAP-HC as an alternative structure learning within MOA, denoted MOA$'$, and denoting by MOA the original version that uses MI. Both versions were tested on two benchmark functions widely used in the EDA's literature: *Royal Road* and *OneMax*, both bit-string optimization tasks, detailed in (27). Each bit-string is modeled in the context of evolutionary algorithms as a chromosome and each bit as a gene. In the Royal Road problem, the variables are arranged in groups of size γ. Its goal is to maximize the number of 1s in the string, but adding γ to the fitness count only when a group has all 1s, otherwise adding 0. For example, in the case of $\gamma = 4$, an individual 111110011111 is separated in the groups [1111] [1001] [1111], and only the first and third groups contributes 4 to the fitness count, which in the example equals 8. The underlying independence structure therefore contains cliques of size γ, one per group. In our experiments we used $\gamma = 1$ and $\gamma = 4$. The former is known in the literature as *OneMax*. In the example, the fitness is 10 for OneMax. Clearly, the optimal individual for both problems is 111111111111.

In the experiments, MOA is iterated for 1000 generations or until the optima is reached, whatever happened first. For several runs differing in the initial (random) population, we measured the *success rate* as the fraction of times the optima is found. A commonly used measure of performance in EDAs is the *critical population size* $\mathcal{D}^*$; the minimum population size for which the success rate is 100%. Smaller $\mathcal{D}^*$ values have a double benefit: (i) fewer fitness evaluations for reaching the optima, and (ii) faster distribution estimation. We report $\mathcal{D}^*$ and the number of fitness evaluations required for that population size, denoted f^*. More robust algorithms are expected to require smaller D^* and f^* values. To measure $\mathcal{D}^*$ in Royal Road and OneMax, each version of MOA was run 10 times for each of the population sizes $\mathcal{D} = \{50, 100, 200, 400, 800, 1600, 3200\}$. Then, for that $\mathcal{D}^*$, we report the average and standard deviation of f^* on each of those runs. In all the experiments the population is truncated with a selection size of 50% and an elitism of 50%; used for preventing diversity loss. In MOA, the parameter k was set to 3 and 1 in Royal Road and OneMax, respectively.

Results are presented in Table II for Royal Road, and OneMax. For both algorithms, MOA and MOA$'$, the table report values of $\mathcal{D}^*$, and the average and standard deviation (in parenthesis) of f^*, for increasing problem sizes $n = 16, 32, 64, 92, 120$ for Royal Road and $n = 15, 30, 60, 90, 120$ for OneMax. Results show that for f^*, MOA$'$ always outperforms MOA; while for $\mathcal{D}^*$, it is always equal or lower. For Royal Road, the larger improvement is for $n = 92$ where MOA$'$ requires 75% fewer fitness evaluations f^* and $\mathcal{D}^*$ is halved. For OneMax, the larger improvement is for $n = 60$ where MOA$'$ requires 80% fewer fitness evaluations f^* and $\mathcal{D}^*$ is reduced to a quarter.

An interpretation of these results is that IBMAP-HC estimates better the distribution. To confirm this hypothesis we compared the structures learned by the two algorithms over the same synthetic datasets considered in the previous section. For $n = 75$, $\mathcal{D} = 100$ and $\tau = 2$ the HDs of MI and IBMAP-HC were 132, and 75, respectively. For $\tau = 4$ they were 233 and 143, respectively; and for $\tau = 8$, 395 and 388, respectively. These results show clearly that the quality of IBMAP-HC indeed outperforms that of MI. Finally, we highlight that the efficiency of IBMAP-HC allowed it to be run in large problems up to 120 genes in size, estimating the structure over many generations.

V. CONCLUSIONS AND FUTURE WORK

This paper introduces a novel independence-based, maximum-a-posteriori approach for learning the structure of MNs; and IBMAP-HC, an efficient instantiation of IBMAP. Unlike traditional SB algorithms, our method follows an IB strategy for getting the MAP independences structure from data proposing an IB score. Experiments comparing IBMAP-HC against IB and SB representative algorithms indicate that our method improves in most cases over the IB competitors with equivalent runtimes, and obtains comparable Hamming distances than the chosen competitor SB algorithm but with orders of magnitude faster runtimes. IBMAP-HC was also tested in a practical, challenging setting: Estimation of Distribution algorithms, resulting in faster convergence to the optimum than a state-of-the-art Markov network EDA algorithm, for the selected benchmark functions. The results, although conclusive in favor of IBMAP-HC, could be reinforced by comparing it against stronger competitors of the SB approach. As future work we believe it is worthwhile exploring alternative instantiations of the IBMAP approach (closure and search). Also, better qualities are expected if the approximation of independence between tests could be relaxed. Finally, the current approach may be extended to tackle the more challenging problem of model learning (structure plus parameters).

Acknowledgements

This work was funded by the scholarship program of the *Universidad Tecnológica Nacional* and the *National Fund for Scientific and Technological Promotion* (FONCyT) of Argentina. Special thanks to Roberto Santana and Siddartha Shakya for their help and support while implementing EDAs.

References

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

[2] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Inc., 1988.

[3] P. Spirtes, C. Glymour, and R. Scheines, *Causation, Prediction, and Search*, ser. Adaptive Computation and Machine Learning Series. MIT Press, 2000.

[4] A. Agresti, *Categorical Data Analysis*, 2nd ed. Wiley, 2002.

[5] D. Margaritis, "Distribution-free learning of Bayesian network structure in continuous domains," in *AAAI*, 2005.

[6] D. Margaritis and F. Bromberg, "Efficient markov network discovery using particle filter," *Comp. Intel.*, vol. 25, no. 4, pp. 367–394, 2009.

[7] C. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, and X. Koutsoukos, "Local causal and markov blanket induction for causal discovery and feature selection for classification part ii: Analysis and extensions," *JMLR*, vol. 11, pp. 235–284, March 2010.

[8] C. F. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, and X. Koutsoukos, "Local causal and markov blanket induction for causal discovery and feature selection for classification part i: Algorithms and empirical evaluation," *JMLR*, vol. 11, pp. 171–234, March 2010.

[9] D. Margaritis and S. Thrun, "Bayesian network induction via local neighborhoods," in *NIPS*, 2000.

[10] F. Bromberg, D. Margaritis, and H. V., "Efficient markov network structure discovery using independence tests," *JAIR*, vol. 35, pp. 449–485, July 2009.

[11] F. Barahona, "On the computational complexity of Ising spin glass models," *Journal of Physics A: Mathematical and General*, vol. 15, no. 10, pp. 3241–3253, 1982.

[12] T. Minka, "Algorithms for maximum-likelihood logistic regression," *Statistics Tech Report*, vol. 758, 2001.

[13] W. Cochran, "Some methods of strengthening the common χ^2 tests," *Biometrics*, vol. 10, pp. 417–451, 1954.

[14] C. Sutton and A. Mccallum, "Piecewise Training for Undirected Models," in *UAI*, 2005.

[15] Y. Yu and Q. Cheng, "MRF parameter estimation by an accelerated method," *Pattern Recogn. Lett.*, vol. 24, no. 9-10, pp. 1251–1259, 2003.

[16] S.-I. Lee, V. Ganapathi, and D. Koller, "Efficient structure learning of Markov networks using L1-regularization," in *NIPS*, 2006.

[17] W. Lam and F. Bacchus, "Learning Bayesian belief networks: an approach based on the MDL principle," *Computational Intelligence*, vol. 10, pp. 269–293, 1994.

[18] J. Besag, "Efficiency of pseudolikelihood estimation for simple Gaussian fields," *Biometrica*, vol. 64, pp. 616–618, 1977.

[19] J. Pearl and A. Paz, "Graphoids: A graph-based logic for reasoning about relevance relations," 1987.

[20] C. Aliferis, I. Tsamardinos, and A. Statnikov, "HITON, a novel Markov blanket algorithm for optimal variable selection," *AMIA Fall*, 2003.

[21] S. Perkins, K. Lacker, and J. Theiler, "Grafting: Fast, incremental feature selection by gradient descent in function space," *JMLR*, vol. 3, pp. 1333–1356, 2003.

[22] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, "Algorithm 78: L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization," vol. 23, no. 4, pp. 550–560, 1997.

[23] H. Mühlenbein and G. Paaß, "From recombination of genes to the estimation of distributions i. binary parameters." Springer-Verlag, 1996, pp. 178–187.

[24] P. Larrañaga and J. A. Lozano, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Pubs, 2002.

[25] M. Alden, "Marleda: Effective distribution estimation through markov random fields," Ph.D. dissertation, Dept of CS, University of Texas Austin, 2007.

[26] S. Shakya and R. Santana, "A markovianity based optimization algorithm." *TR, Basque Country U.*, 2008.

[27] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA, USA: MIT Press, 1998.