Strategies for piecing-together Local-to-Global Markov network learning algorithms

F. Schlüter, F. Bromberg, L. Abraham

{federico.schluter, fbromberg, leandro.abraham}@frm.utn.edu.ar
Laboratorio DHARMa de Inteligencia Artificial.
Dept. Sistemas de Información, Facultad Regional Mendoza,
Universidad Tecnológica Nacional, Mendoza, Argentina

Abstract. We introduce in this work a set of strategies for improving the piecing-together step in Local-to-global Markov networks structure learning algorithms. For Markov networks, Local-to-global algorithms decompose the problem of learning a complete independence structure with n variables into n independent Markov blanket learning problems. On a second step these algorithms piece-together all the learned Markov blankets into a global structure using an "OR rule". Insufficient data may result in incorrect learning of Markov blankets, with conflicts in their decision on edge inclusion when, for two variables X and Y, X is in the blanket of Y, but Y is not in the blanket of X. In such cases the "OR rule" always decides to add the edge, making mistakes when such edge does not exist. Our contribution are alternative strategies. The first alternative is based on the "AND rule" which proposes to add an edge between two variables X and Y to the global structure if they mutually belongs to its respective Markov blankets. The other alternative rule is based on the probability of the edges and aims to solve an inconsistency by comparing the probability of edge existence with the probability of edge absence, and taking the more probable for deciding to add or remove such edge. At the end of the work we show that inconsistencies are an important source of errors in this algorithms, and demonstrate empirically interesting improvements in the quality of learned structures, using this new piecing-together alternatives instead of the basic "OR rule".

Keywords: Markov networks, structure learning, independence-based, local-to-global learning.

1 Introduction

Graphical models are a well-established formalism for representing compactly joint probability distributions. Markov networks, together with the well-known Bayesian networks are widely accepted types of graphical models [17]. Such models are composed of two parts: a qualitative, and a quantitative one. The qualitative component is encoded in a graph G representing structural information about the domain of a problem in the form of independence relationships between variables, known as the *independence structure*. The structure G contains

a node for each random variable of the domain, and a set of undirected (for Markov networks) or directed (for Bayesian networks) edges for encoding conditional independences between such variables. The quantitative component is a set of numerical parameters $\boldsymbol{\theta}$, usually in the form of tables with real values for quantifying the relationships in G. Since the problem of building G and $\boldsymbol{\theta}$ may be difficult and time-consuming by eliciting opinions from domain experts, and the increasing availability of digital data, the challenge of eliciting graphical models from data has been under intense work in the last two decades. There are two main approaches in the literature for learning graphical models from data: *Score-Based* and *Independence-Based* approaches.

Score-based algorithms [21, 19, 13] approach the problem as an optimization on the space of complete models (G and $\boldsymbol{\theta}$), looking for the one with maximum *score*. Some example scores in the literature are *maximum likelihood*, *minimum description length* [12], and *Pseudo-likelihood* [6, 21], among others. An advantage of the score-based algorithms is their resiliency to data scarceness. However, for the case of Markov networks, its learning is at the expense of important computational costs. For each structure during the search, they require the estimation of $\boldsymbol{\theta}$; computationally intensive for requiring an expensive inference step [5]. Such estimation is an NP-hard problem, solved in practice through a data-intensive numerical algorithm [16]. Also, for avoiding over-fitting, many of these methods use a regularization term adding an extra hyper-parameter, whose best value has to be found empirically (e.g., running the training stage for several values of the hyper-parameter, potentially with cross-validation), yet another cause of inefficiency.

Independence-based (also known as constraint-based) algorithms first learn the qualitative part of the model (that is, G), and then, if the complete model is required, estimate the quantitative part $\boldsymbol{\theta}$ for the given structure G. Such algorithms perform a succession of *statistical independence tests* [18] for learning G. Each independence test consults the data for responding to a query about the conditional independence between some input random variables X and Y, given some conditioning set of variables $\mathbf{Z}$, resulting in an independence assertion, denoted $I(X, Y \mid \mathbf{Z})$, or a dependence assertion, denoted $\neg I(X, Y \mid \mathbf{Z})$. When data is scarce, these assertions may be incorrect, i.e., opposite to their true value in the underlying model. Examples of independence tests used in practice are Mutual Information [11], Pearson's χ^2 and G^2 [1], the Bayesian test [14], and for continuous Gaussian data the *partial correlation* test [18]. To learn G, independence-based algorithms proceed iteratively, deciding on each iteration the test to be executed based on the independences learned so far, and then, discarding all the structures inconsistent with the independence outcome of the test, until a single structure is left. Independence-based algorithms have several advantages. First, they can learn G without estimating $\boldsymbol{\theta}$ (contrary to score-based algorithms, as explained before), reaching polynomial complexities in the number of statistical tests in some cases. Another advantage is that they are amenable to proof of correctness when some assumptions hold: positivity of the distribution, the underlying distribution is a Markov network, and sta-

tistical tests are correct. The latter assumption is violated unless the dataset used for learning is sufficiently large. Unfortunately, statistical tests reliability degrades exponentially with the amount of variables involved (for some fixed size of 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 test). For example, for the χ^2 test [10] recommends that the test be deemed unreliable if more than 20% of these cells have an expected count of less than 5 data points. For Bayesian networks learning, the independence-based approach has been used by the well known SGS and PC algorithms [18], and the family of structure learning algorithms generalized by the Local-to-Global Learning (LGL) framework of [3]. LGL is a framework for producing structure learning algorithms from algorithms that learn the local neighborhood of variables in Bayesian networks (according to the Generalized Local Learning framework of [2]). This particular type of independence-based algorithms learns a global structure with n variables dividing the problem in n independent local neighborhood learning problems, and constructs a global undirected structure by *piecing-together* the local neighborhoods. In a third step, it orients the edges of such structure. For Bayesian networks, a possible local neighborhood of a variable X is given by its *Markov blanket* $\mathbf{MB}(X)$. In terms of independences, $MB(X)$ is a minimal subset of the variables in the universe $\mathbf{V}$ conditioned on which all other variables are probabilistically independent of X (formal definition in the following section). Examples of algorithms for learning it from data are the Grow-Shrink (GS) algorithm [15], and IAMB [20], among others. The independences encoded in a Bayesian networks results in the $MB(X)$ being the set of parent, children, and spouses of X in the directed graph, and the independences encoded in a Markov network results in the $\mathbf{MB}(X)$ being the set of variables directly connected in the undirected graph. In the case of Bayesian networks, it is possible to learn the directed structure from smaller neighborhoods than $MB(X)$: the set of parents (P) and children (C) variables of a node in the network, as exemplified by the HITON-PC algorithm [4] and its corresponding structure learning algorithm HHC [3]. For Markov networks, adaptations of the LGL approach has been proposed based on the GS local learning algorithm, resulting in the GSMN structure learning algorithm [7], and based on an adaptation of the HITON-PC algorithm for Markov networks, resulting in the HHC-MN structure learning algorithm [9].

The remainder of this paper is organized as follows. Section 2 presents an overview of the state-of-the-art LGL algorithms for Markov networks, and introduces the discussion about the problem of the *inconsistencies* that arise when *piecing-together* local neighborhoods of LGL algorithms. In addition, it discusses the main contribution of this work: a set of strategies for improving the piecing-together step in LGL algorithms for Markov networks. Section 3 presents experimental results of a comparison between different LGL algorithms and different strategies for piecing-together. The paper concludes with a summary and possible directions of future work in Section 4.

2 Strategies for piecing together local structures

In this section we explain the outline of a general LGL algorithm for Markov networks, together with an important source of errors in such kind of algorithms: the *inconsistencies*. We also describe three instantiations of such algorithm,used in this work as a testbed for experimentation. Finally, we present the main contribution of this work: novel strategies for improving the piecing-together step in LGL algorithms for Markov networks.

2.1 Local-to-global learning algorithms for Markov networks

Algorithm 1 outlines the local-to-global general strategy for Markov networks. This is an adaptation for Markov networks learning of the HHC algorithm, a state-of-the-art LGL algorithm for Bayesian networks presented in [3], omitting only the last step, in which the edges are oriented. The algorithm starts by dividing the problem in n independent Markov blanket learning problems. Then, the algorithm *pieces-together* all the learned Markov blankets into a global structure using an "OR rule", that is, the global structure is constructed adding an edge between two variables X and Y if and only if $X \in MB(Y)$ OR $Y \in MB(X)$.

Algorithm 1 LGL for Markov networks

1: Learn MB(X) for every variable X in the domain.
2: Piece-together the global structure using an "OR rule".

We now sketch the correctness of this approach. For that, let us first define formally the Markov blanket concept:

Definition 1 (Markov blanket, [17], p.97). *A* Markov blanket *of some random variable $X \in \mathbf{V}$ is the minimal set $MB(X)$ that shields the probabilistic influence of every other variable with X, i.e., the set $MB(X) \subseteq \mathbf{V} - \{X\}$ such that*

$$\forall Y \in \mathbf{V} - \{X\}, Y \notin \mathbf{MB}(X) \Rightarrow I(X, Y \mid \mathbf{MB}(X) - \{Y\}) \tag{1}$$

The fact that blankets of all variables then can be combined to construct the global structure G is proved by **Corollary 2** of [17], p.98, which asserts that the independence structure G of any strictly positive distribution over $\mathbf{V}$ can be constructed by connecting each variable $X \in \mathbf{V}$ to all members of its $\mathbf{MB}(X)$. Formally,

$$\forall Y \in \mathbf{V} - \{X\}, Y \in \mathbf{MB}(X) \Leftrightarrow E(X, Y), \tag{2}$$

where $E(X, Y)$ means that there is an edge between X and Y in the graph G.

2.2 Unreliable tests: solving inconsistencies

It can be proven [17] that whenever tests are correct, an edge (X, Y) exists in the network if and only if $X \in MB(Y)$ and $Y \in MB(X)$. As discussed, tests may be incorrect when data is scarce, resulting in *inconsistencies* in the Markov

blankets. A *Markov blanket inconsistency*, or simply *inconsistency*, between two variables X and Y happens when $Y \in MB(X)$, but $X \notin MB(Y)$, (or vice versa). Figure 1 illustrates inconsistencies. Case (a) shows an undirected graph for $n = 3$ and $\mathbf{V} = \{0, 1, 2\}$, with correct blankets for variables 0, 1, and 2 shown in (b), (c) and (d), respectively. In this figure the directed edges in the graph indicate that a variable belongs to the blanket of the target variable (should not be confused with the directed edges of structures in Bayesian networks). Illustrations (e), (f), and (g) show some example *learned* blankets of variables 0, 1, and 2, respectively. One can observe inconsistencies in these illustrations between variables 0 and 1 ($1 \in MB(0)$ but $0 \notin MB(1)$), and between 0 and 2 ($2 \notin MB(0)$ but $0 \in MB(2)$).

When blanket inclusion $X \in MB(Y)$ is interpreted as edge existence, inconsistencies are clearly a source of error. The "OR rule" solves this by simply deciding edge inclusion, but there is no basis for this assertion.

2.3 Instantiations of LGL algorithms for Markov networks

In this section we describe three instantiations of Algorithm 1: GSMN [7], HHC-MN [9], and LGL-IBMAPHC (introduced in this work). The only difference between them is the Markov blanket learning algorithm they use. GS [15], HITON [4], and IBMAPHC [9] are the algorithms used for learning MB of variables for the GSMN, HHC-MN, and LGL-IBMAPHC algorithms, respectively.

First we explain the HITON and GS algorithms, described in detail in Algorithm 2 and Algorithm 3, respectively. Since they are similar we will describe the features that they have in common. Its input parameters are: X, the target variable for which the Markov blanket is learned; $\mathbf{V}$, the set of all the variables in the domain; D, a dataset used for testing independence; and SIT, a Statistical Independence Test. Both algorithms consist in three phases: initialization, grow and shrink. The initialization phase populates the FIFO queue called OPEN with variables $\mathbf{V} - \{X\}$, sorted by the unconditional dependency between X and each variable, using the statistical test of independence SIT. Each iteration of the grow phase considers adding a variable Y in OPEN to the current set of tentative Markov blanket $TMB(X)$. For symmetry with the shrink phase, this

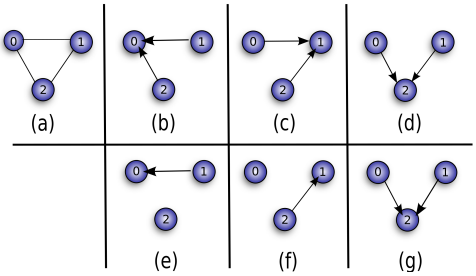

Fig. 1. (a) An example undirected graph for $n = 3$, and its correct MB for the three variables 0, 1 and 2 in (b),(c) and (d), respectively. Directed edges denote membership to the Markov blanket (should not be confused with directed edges of structures in Bayesian networks). Graphs in (e), (f) and (g) show example learned Markov blankets and the inconsistencies that arise: $1 \in MB(0)$ but $0 \notin MB(1)$, and $0 \in MB(2)$ but $2 \notin MB(0)$.

Algorithm 2 HITON (X, V, D, SIT)

```
1: /* Initialization phase */
2: TMB(X) ← ∅
3: OPEN ← V − {X}
4: Sort in ascending order the variables in OPEN according to SIT(⟨X, Y, ∅⟩, D).
5: while OPEN ≠ ∅ do
6:     /* Grow phase */
7:     Y ← pop(OPEN)
8:     add Y to TMB(X)
9:     if ∃Z ⊆ TMB(X) − {Y} : SIT(⟨X, Y, Z⟩) then
10:        remove Y from TMB(X)
11:    /* Shrink phase */
12:    for each variable Y ∈ TMB(X) do
13:        if ∃Z ⊆ TMB(X) − {Y} : SIT(⟨X, Y, Z⟩)) then
14:            remove Y from TMB(X)
15: return TMB(X)
```

proceeds by first adding it to $TMB(X)$ and testing if it should be eliminated using an elimination criteria. The shrink phase removes each false positive Y in $TMB(X)$ using the same elimination criteria. The first difference between algorithm 2 and 3 is the elimination criteria used. As shown in lines 9 and 13 of the Algorithm 2 this algorithm checks for independence of the variables X and Y, conditioned on its tentative Markov blanket or any of its subsets. On the other hand the Algorithm 3 on lines 9 and 13 checks for independence of the variables X and Y, conditioned on its tentative Markov blanket. The second difference between them is the order of execution of the Shrink phase. As we can see, in the Algorithm 2 it is executed interleaved into the Grow phase (line 12). The Algorithm 3, however executes this phase outside the loop of the Grow phase (line 12). As explained before, HITON and GS learns the exact Markov blanket when tests are correct, otherwise, HITON has been proven empirically to be more robust [2].

The third instantiation of the LGL generic approach of Algorithm 1 considered for experimentation is LGL-IBMAPHC, that learns the MB using a variation of the IBMAPHC algorithm [9]. IBMAPHC is a recent algorithm for learning the complete independence structure, with comparable results to the HHC-MN algorithm. It performs a hill-climbing search over the space of graphs G looking for the one which maximizes the IB-score, a score of the posterior probabilities of graphs $\Pr(G \mid D)$. The LGL-IBMAPHC proposed here constructs the global structure following the steps of Algorithm 1, and learning the Markov blanket for all the variables with the IBMAPHC algorithm, adapted for Markov blankets learning, instead of global structure learning. Such simple adaptation is done by adapting the score of total structures to the score of Markov blankets $\Pr(MB(X) \mid D)$, and adapting the hill-climbing search for using the search space of Markov blankets, instead of graphs.

The IB-score for complete structures is computed using a set of independence assertions which are sufficient for determining G completely,called a *clo-*

Algorithm 3 GS (X, V, D, SIT)

```
/* Initialization phase */
TMB(X) ← ∅
OPEN ← V − {X}
Sort in ascending order the variables in OPEN according to SIT(⟨X, Y, ∅⟩, D).
while OPEN ≠ ∅ do
    /* Grow phase */
    Y ← pop(OPEN)
    add Y to TMB(X)
    if SIT(⟨X, Y, TMB(X) − {Y}⟩) then
        remove Y from TMB(X)
/* Shrink phase */
for each variable Y ∈ TMB(X) do
    if SIT(⟨X, Y, TMB(X) − {Y}⟩) then
        remove Y from TMB(X)
return TMB(X)
```

sure $\mathcal{C}(G)$, and denoted by $\mathcal{C}(G) = \left\{T_i = t_i^G\right\}_{i=1}^{c}$, $c = |\mathcal{C}(G)|$, where $t_i^G \in \{d, \neg d\}$ denotes the value that the independence random variable T_i takes in the graph G, that is, whether the triplet corresponding to T_i is independent or not in G (see details in the reference). The posterior probability of G is computed by the IB-score as follows:

$$\text{IB-score}(G) = \Pr(\mathcal{C}(G) \mid \mathcal{D}) \simeq \prod_{(T=t^G)\in\mathcal{C}(G)} \Pr(t^G \mid \mathcal{D}), \tag{3}$$

where the posteriors $\Pr(T = t^G \mid \mathcal{D})$ can be computed by the Bayesian test of independence of Margaritis [14]. The closure set for the total structure $\mathcal{C}(G)$ utilized is based on Markov blankets, defined as follows:

$$\mathcal{C}(G) = \bigcup_{x\in\mathbf{V}} C_x(G) \tag{4}$$

$$\begin{aligned} C_x(G) = &\left\{\neg I(X, Y \mid MB(X) - \{Y\}) \;\middle|\; Y \in MB(X)\right\} \\ &\bigcup \left\{I(X, Y | MB(X) - \{Y\}) \;\middle|\; Y \notin MB(X)\right\} \end{aligned} \tag{5}$$

containing, for a given structure G, an amount of $(n-1)$ assertions for each of the n variables, and this way determining completely the structure. For computing the IB-score for $MB(X)$, the closure is defined as a simplification only for the variable X, containing its corresponding $(n-1)$ assertions.

2.4 Strategies for piecing-together local structures

This section presents the main contribution of this work, a set of alternative strategies for piecing-together Markov blankets under the existence of inconsistencies. The hypothesis is that LGL algorithms producing large amount of

inconsistencies are prone to a larger reduction in erroneous edges when a good piecing-together strategy is followed.

Currently the literature proposes only the "OR rule" (to the best of the authors knowledge), which under an inconsistency, always adds an edge (the OR is always satisfied for an inconsistency). As a first alternative rule, we propose the "AND rule", which adds an edge between variables X and Y to the global structure only if $X \in \mathbf{MB}(Y)$ and $Y \in \mathbf{MB}(X)$. That is, under an inconsistency, it always decides independence. Such rule seems to be more robust for adding correct dependencies, since it only adds an edge between two variables when such dependency is learned in both directions. On the other hand, it may produce larger amounts of incorrect independences.

In addition, we propose a 'soft' strategy we call the "EP rule" (Edges Probability rule). Under an inconsistency, it decides to add an edge only if the probability of edge existence $Pr(E(X,Y))$ is larger than the probability of edge non-existence $Pr(\neg E(X,Y))$, thus its name.

We now show that the probabilities of edge existence and non-existence can be computed by the probabilities of certain independences. For that, let us first prove the following Lemma,

Lemma 1.

$$\neg I(X, Y \mid \mathbf{MB}(X) - \{Y\}) \Leftrightarrow Y \in \mathbf{MB}(X) \tag{6}$$

Proof. The ($\Rightarrow$) direction follows directly from the counter-positive of the definition of blankets in Eq. (1). The opposite direction ($\Leftarrow$) is more involved, and has been proven by Lemma 3 of [8].

Assuming the inconsistency we are solving is $Y \in \mathbf{MB}(X)$ but $X \notin \mathbf{MB}(Y)$ (equivalent conlusions for case $Y \notin \mathbf{MB}(X)$ but $X \in \mathbf{MB}(Y)$ may be obtained by symmetry), a direct Corollary of Eqs. 2 and 6 is that

$$\neg I(X, Y \mid \mathbf{MB}(X) - \{Y\}) \Leftrightarrow E(X, Y) \tag{7}$$

and therefore

$$\begin{aligned} Pr(E(X,Y)) &= Pr(\neg I(X, Y \mid \mathbf{MB}(X) - \{Y\})) \\ Pr(\neg E(X,Y)) &= Pr(I(X, Y \mid \mathbf{MB}(Y) - \{X\})). \end{aligned} \tag{8}$$

In summary, the decision of edge or no edge between variables X and Y when the algorithm finds that $Y \in \mathbf{MB}(X)$ but $X \notin \mathbf{MB}(Y)$ is taken by comparing probabilities $Pr(\neg I(X, Y \mid \mathbf{MB}(X) - \{Y\}))$ and $Pr(I(X, Y \mid \mathbf{MB}(Y) - \{X\}))$. If the former is greater, the EP rule decides there is an edge between X and Y, otherwise, it decides no edge. To compute these probabilities of independence assertions we use the Bayesian test of Margaritis [14].

3 Experimental results

This section shows a comparison of the quality of structures learned by LGL algorithms GSMN, HHC-MN, and LGL-IBMAPHC, when the different piecing-together strategies OR, AND, and EP are used for dealing with inconsistencies. Results show empirically the benefit of the proposed strategy EP.

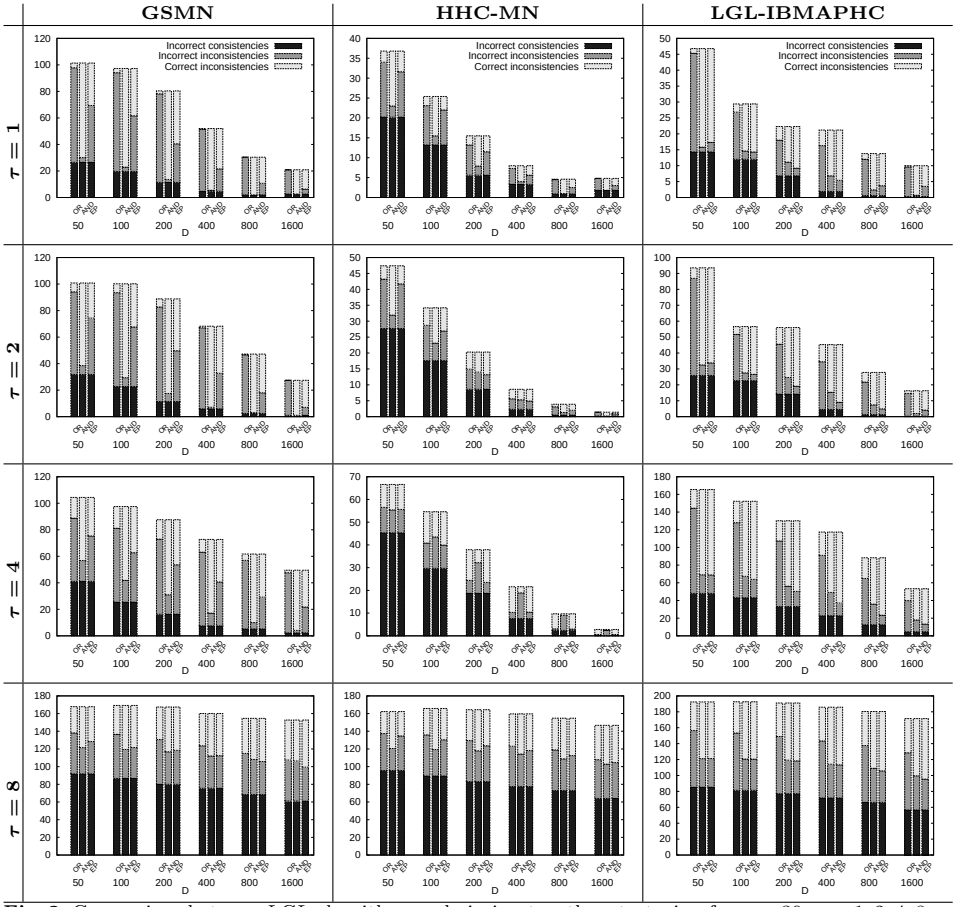

Fig. 2. Comparison between LGL algorithm, and piecing-together strategies, for $n = 30$, $\tau = 1, 2, 4, 8$ and $D = 50, 100, 200, 400, 800, 1600$

The artificial datasets used to test the algorithms and strategies were generated using a Gibbs sampler, sampling from known random models. Knowing the solution structure allows a systematic and controlled study of the results. The models were generated for domains of $n = 30$ random binary variables. For this domain, 10 random networks were generated, each of them with increasing number of neighbors per node $\tau = \{1, 2, 4, 8\}$, by selecting the first $n\tau/2$ edges of a random permutation of all variables pairs. Given these networks, datasets with increasing number of data points $D = \{50, 100, 200, 400, 800, 1600\}$ were sampled for each (n,τ) configuration.

Each algorithm results in some inconsistencies as well as consistencies; each of which may decide correctly or not on the corresponding edge by the piecing-together strategy. These results in four quantities of interest: *correct consistencies*, *correct inconsistencies*, *incorrect consistencies*, and *incorrect inconsistencies*. Figure 2 reports the last three in white, black, and grey bars, respectively; for increasing dataset sizes. Note that the sum of the last two (black and grey

bars) equals the Hamming distance (HD), while the sum of second and fourth (white and grey bars) corresponds to the total number of inconsistencies . These figures therefore gives us two important metrics: (i) the HD as a measure of error, and (ii) the proportion of inconsistencies correctly decided by each strategy (grey bar vs sum of grey and white bars). The figure shows that the "AND rule" presents in most cases smaller values of incorrect inconsistencies than the values obtained by the "OR rule" (except some cases when using HHC-MN, $\tau = 4$ and $D \geq 100$). Besides, in all cases the incorrect inconsistencies values of the "EP rule" are smaller than the obtained by the "OR rule". Making a comparison between the "AND rule" and the "EP rule", the "AND rule" is better than the "EP rule" since the amount of incorrect inconsistencies obtained by "EP rule" are always higher, except some particular cases where the "EP rule" is better (that is, for HHC-MN, LGL-IBMAPHC and $\tau = 4$). These results show that it is possible to reduce the total HD of the structure, using different piecing-together strategies than the simple "OR rule" for LGL algorithms.

Figure 3 reports complementary metrics: *precision* and *recall* measures. On one hand, precision shows how good was the algorithm to learn correct edges. It quantifies the proportion of edges learned correctly among those learned (true positives / true positives + false positives, meaning by positive a dependency or edge). It is a value between 0 and 1, with 1 being the best possible precision, i.e., all edges learned were correct. On the other hand, recall reports how many of the true edges were learned correctly (true positives / true positives + false negatives). Also a value between 0 and 1, with 1 meaning that all the edges of the solution structure were correctly learned.

These results show again that the strategies presented in this work make important improvements over the precision and recall of the "OR rule". Focusing our analysis over the "AND rule" and the "EP rule" , the precision obtained by "AND rule" is always higher or equal than the "EP rule", and always both strategies improves the precision of the "OR rule". However, the recall values of the "AND rule" presents lower values than those for "EP rule". Furthermore, the recall of "EP rule" is in most cases better or equal to those values for the "OR rule"(except some exceptions for GSMN and LGL-IBMAPHC with $\tau = 4, 8$). Therefore, the structures learned with the "AND rule" introduce more false independences than the ones learned by the "EP rule" as we can see according to the recall results. The fact of adding false dependencies, is not as bad as adding false independences, because false dependences result on a more complex but still correct model; however a false independence results on a more simple but erroneous model.In summary, the "EP rule" is better than the "AND rule" because using the latter may introduce really important errors in the resulting model by adding false independences.

4 Conclusions

In this work we made an analysis of the inconsistencies problem of local-to-global Markov networks structure learning algorithms, and we introduced two

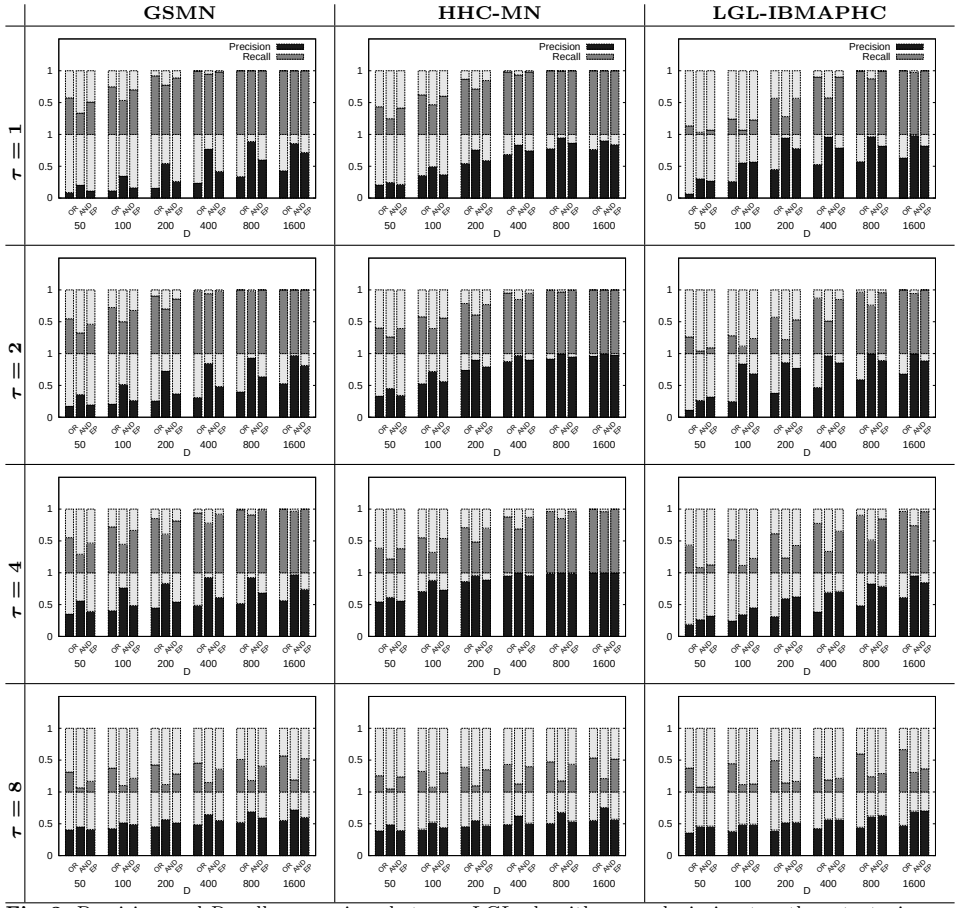

Fig. 3. Precision and Recall comparison between LGL algorithms, and piecing-together strategies, for $n = 30$, $\tau = 1, 2, 4, 8$ and $D = 50, 100, 200, 400, 800, 1600$

new strategies for the piecing-together step. We proved empirically that quality of learned structures can be improved by using different strategies than the basic "OR rule" to solve the inconsistencies. For this, we performed several experiments over three local-to-global algorithms combined with three different piecing-together strategies discussed in this work. The results obtained confirm our hypothesis showing important quality improvements on the learned structures using the presented strategies ("AND", "EP") over the "OR" strategy. Such results show also that inconsistencies are an important source of error in local-to-global algorithms. In some cases the "AND rule" seems to be a better choice than the "EP rule", but it is important to remember that using this alternative may introduce errors in the model by adding false independences. Because of that, we recommend as the best choice the "EP rule" because it is better than the OR alternative in all cases, and it introduces less errors in the model than the "AND rule". Future work could be focused in the development or improvement of the local-to-global Markov networks structure learning algo-

rithms so that they learn the blankets with a reduced number of inconsistent edges. Another line of work consists in further improving the "EP" strategy, or the proposal of alternative strategies.

5 Acknowledgements

This research has been funded by the FONCyT (Argentinean National Fund for Technology and Science) and the Universidad Tecnológica Nacional.

References

[1] A. Agresti. *Categorical Data Analysis*. Wiley, 2nd edition, 2002.
[2] 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 i: Algorithms and empirical evaluation. *JMLR*, 11:171–234, March 2010.
[3] 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*, 11:235–284, March 2010.
[4] C. Aliferis, I. Tsamardinos, and A. Statnikov. HITON, a novel Markov blanket algorithm for optimal variable selection. *AMIA Fall*, 2003.
[5] F. Barahona. On the computational complexity of Ising spin glass models. *Journal of Physics A: Mathematical and General*, 15(10):3241–3253, 1982.
[6] J. Besag. Efficiency of pseudolikelihood estimation for simple Gaussian fields. *Biometrica*, 64:616–618, 1977.
[7] F. Bromberg, D. Margaritis, and H. V. Efficient markov network structure discovery using independence tests. *JAIR*, 35:449–485, July 2009.
[8] F. Bromberg and F. Schlüter. Efficient Independence-Based MAP Approach for Robust Markov Networks Structure Discovery. 2011.
[9] F. Bromberg, F. Schlüter, and A. Edera. Independence-based MAP for Markov networks structure discovery. *International Conference on Tools with Artificial Intelligence 2011*, Submitted on July 2011.
[10] W. G. Cochran. Some methods of strengthening the common χ^2 tests. *Biometrics*, 10:417–451, 1954.
[11] T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley-Interscience, New York, NY, USA, 1991.
[12] W. Lam and F. Bacchus. Learning Bayesian belief networks: an approach based on the MDL principle. *Computational Intelligence*, 10:269–293, 1994.
[13] S.-I. Lee, V. Ganapathi, and D. Koller. Efficient structure learning of Markov networks using L1-regularization. In *NIPS*, 2006.
[14] D. Margaritis. Distribution-free learning of Bayesian network structure in continuous domains. In *Proceedings of AAAI*, 2005.
[15] D. Margaritis and S. Thrun. Bayesian network induction via local neighborhoods. In *Proceedings of NIPS*, 2000.
[16] T. Minka. Algorithms for maximum-likelihood logistic regression. Technical report, Dept of Statistics, Carnegie Mellon University, 2001.
[17] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Inc., 1988.
[18] P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. Adaptive Computation and Machine Learning Series. MIT Press, 2000.
[19] C. Sutton and A. Mccallum. Piecewise Training for Undirected Models. In *UAI05*.
[20] I. Tsamardinos, C. F. Aliferis, and A. Statnikov. Algorithms for large scale Markov blanket discovery. In *FLAIRS*, 2003.
[21] Y. Yu and Q. Cheng. MRF parameter estimation by an accelerated method. *Pattern Recogn. Lett.*, 24(9-10):1251–1259, 2003.