Summarizing and visualizing a set of bayesian networks with quasi essential graphs

Hoai-Tuong Nguyen, Philippe Leray, Gérard Ramstein

Summarizing and visualizing a set of Bayesian networks with Quasi Essential Graphs

Hoai-Tuong Nguyen, Philippe Leray, and Gérard Ramstein

Nantes Atlantique Computer Science Lab LINA UMR 6241
Knowledge and Decision Team
La Chantrerie - rue Christian Pauc - BP 50609 - 44360 Nantes Cedex 3, France
(E-mail: first-name.last-name@univ-nantes.fr)

Abstract. Many learning methods now generate a set of models in order to improve robustness. Evaluating for instance the quality of a set of Bayesian networks is quite usual for estimating separately the quality of each model and for summarizing these results. Visualizing the outcomes are a more complex task. We propose in this work an approach based on an inverse principle. Firstly, we build the Quasi Essential Graph (QEG), "most" representative of the whole set. Then, we apply the usual quality operators for this new object. This paper describes the notion and properties of Quasi Essential Graph. An algorithm for its extraction is proposed, as well as a graphical metaphor for its visualization. A toy example is finally given for the sake of illustration.
Keywords: Bayesian Network, Structure Learning, Visualization, Quasi Essential Graph.

1 Introduction

Bayesian networks (BNs) are probabilistic graphical models which have been widely used for prediction or classification tasks in various domains. In the first applications, the BN structure was causally defined by expert knowledge. Then, algorithms were proposed in order to learn the BN structure from observational data.

Learning the structure of a Bayesian network from data is NP-complete [CGH94], so heuristics have to be used in order to find one good local optimum. Furthermore, in many real applications where these models are used, the sample size of available data is much less than the number of observed variables. For these reasons, many existing structure learning algorithms propose using evolutionary approaches [LPY$^+$96,DBC07,MC07,AdF07,WY10] or bootstrap approaches [FGW99,RB05] in order to learn a set of candidate structures instead of one.

Evaluating the quality of one structure obtained with a classical structure learning algorithm is quite simple. When the structure learning has to be validated, the learnt model can be compared to the expected one by classical measures on the graphs (edit distance, ...) or on the corresponding probability distributions (Kullback Leiber divergence). Otherwise, the ability

to represent the learning data can be measured by computing one score, approximation of the marginal likelihood used in BN structure learning [CH96]. Finally, the obtained graph can be visualized with the help of classical graph placement algorithms.

One major difficulty for graph comparison approaches is the Markov equivalence : some edges can be inverted without changing the underlying independence model which has been identified with the learning dataset, so directly comparing the learnt graph and the expected one is not appropriate. One solution is using one property of Markov equivalence: all the equivalent graphs can be summarized by a partially directed graph named complete PDAG, essential graph or pattern. The right solution to graph comparison is then comparing the essential graphs corresponding to the learnt graph and the expected one.

Our problem here is quite more complex: we do not want to evaluate the quality of one graph, but the quality of a set of BNs obtained for instance by bootstrap or by evolutionary approaches. One first and trivial solution is estimating the "mean" quality of this set by using one quality operator (edit distance, KL divergence, score, ...) for each model and then estimate the mean or visualizing the distribution of these results. Graph visualization is not appropriate when the number of BNs in the set is too high.

We propose in this work an approach based on an inverse principle: we first find a "most" graphical representative model for the set of BNs, the Quasi Essential Graph (QEG), and then we apply the usual quality operators for this new object. One disadvantage of this approach is that this QEG will store more information than a simple graph, so these usual quality operators will have to be redefined for this new object. One main advantage of this QEG, and inspiration of this work, is that this object will be easily visualized given a graphical metaphor proposed here.

After giving some preliminary definitions about Bayesian network in section 2, we will then define, in section 3, the notion of Quasi Essential Graph, some interesting properties, an algorithm to built it from a set of BNs and a graphical metaphor for visualizing it. We will then illustrate this QEG on a toy example.

2 Bayesian Networks

2.1 Definition

A *Bayesian network* (BN) $\langle V, G, \theta \rangle\rangle$ is defined by $V = \{X_1, \ldots, X_n\}$, a set of observable discrete random variables, a directed acyclic graph (DAG) G, where each node represents a variable from V, and a set of conditional probability distributions (CPD) $\theta = \{P(X_i|Pa(X_i))\}$ for each variable X_i from V conditionally to its parents $Pa(X_i)$ in the graph G.

When some properties named Markov properties are respected (cf. [Nea03] for more details about these properties), the BN graph describes a set of (con-

ditional) dependence or independence statements which lead to the factorization of the joint probability distribution $P(X_1, \ldots, X_n) = \prod_{i=1}^{n} P(X_i|Pa(X_i))$.

2.2 Markov Equivalence

This relation between DAGs and independence models is not bijective. Two DAGs describing the same (conditional) dependence or independence statements are said *Markov equivalent*.

A set of Markov equivalent DAGs is named a *Markov equivalence class*. All DAGs in the same equivalence class share some graphical properties which can be summarized into a partially directed acyclic graph named CPDAG (complete PDAG) [Chi02], essential graph (EG) [AMP95] or pattern [SM95].

An edge is *compelled* if its orientation is the same in all DAGs of an equivalence class. An edge is *reversible* if it is not compelled. The CPDAG is then a partially directed graph defined by a list of directed edges corresponding to every compelled edge in the equivalence class and a set of undirected edge corresponding to every reversible edge in this class. One consequence is that all equivalent DAGs have the same skeleton and the same compelled edges [VP91].

[DT92] propose an algorithm for generating one DAG consistent with a given Essential Graph, i.e. belonging to the given equivalence class. [Chi02] provides an efficient algorithm for determining the Essential Graph of a given DAG.

3 Quasi Essential Graphs

We have seen in the previous section that a set of equivalent DAGs can be exactly summarized by the Essential Graph associated to the equivalent class. We define here the notion of Quasi Essential Graph (QEG) in order to approximately summarize any set of DAGs. We then give some interesting properties of QEG, and describe one algorithm to build it from a given set of DAGs and one graphical metaphor to visualize it.

3.1 Definition

A *Quasi Essential Graph* $\langle V, G, w_u, w_a \rangle$ is a weighted graph defined by $V = \{X_1, \ldots, X_n\}$, a set of observable discrete random variables, a DAG G, where each node represents a variable from V, a set of weights w_u associated to each (undirected) edge in G skeleton, and a set of weights w_a associated to arrows of each directed edge in G.

One QEG Q will summarize a set of BNs $\mathcal{B}$, for a given threshold $\alpha > 0.5$ (ensuring Q being acyclic), if and only if the three following conditions are true :

(C_0) Q and all the DAGs in $\mathcal{B}$ are defined for the same set of variables V,

(C_1) two vertices X_i and X_j are adjacent in Q iff their probability of being adjacent in $\mathcal{B}$ is equal to $w_u(X_i$–$X_j)$ and greater than α,

(C_2) a directed edge $X_i \rightarrow X_j$ exists in Q iff its probability of being present in $EG(\mathcal{B})$, i.e. the set of equivalent graphs associated to $\mathcal{B}$, when X_i and X_j are adjacent is equal to $w_a(X_i \rightarrow X_j)$ and greater than α.

3.2 Properties

We propose our QEG approach for summarizing a set of BNs. Quasi Essential Graph summarize the information contained in the given set $\mathcal{B}$: empty graph when there is no information (random distribution) or essential graph when information are consistent (all DAGs in the same equivalence class, or with small perturbations around a given graph).

QEG for BNs in the same equivalence class
The QEG summarizing a set $\mathcal{B}$ of *equivalent* BNs is defined by a partially DAG G equal to the unique Essential Graph associated to $\mathcal{B}$ and weights w_u and w_a equal to 1.

Proof is quite simple: if all DAGs in $\mathcal{B}$, they have the same skeleton, so $w_u = 1$ for all edges of this skeleton, and they have the same compelled edges (cf. section 2.2) so $w_a = 1$ for all the corresponding arrows and G will be defined by this common skeleton and arrows, which is the definition of the essential graph of an equivalence class.

QEG for random BNs from a uniform distribution in DAG space
The QEG summarizing a (large) set $\mathcal{B}$ of BNs generated according to a uniform distribution in DAG space is defined by a partially DAG G equal to the undirected graph completely connected G_c or to the empty graph $G_\emptyset$ depending to the value of α, weights w_u equal to 1 if $G = G_c$, 0 otherwise and $w_a = 0$.

Our proof is quite intuitive. There are two directed ways of connecting two vertices (from left or from right) for three configurations (the last one is the disconnection). Even if one removes all configurations leading to a cycle (impossible in directed acyclic graph), the probability of the two vertices being connected will be below 2/3 and greater than 0.5. So all the vertices will be present in G (leading to $G = G_c$) if α is low. If α increases, all the vertices will be discarded and $G = G_\emptyset$.

3.3 Determination

Algorithm *DAGsToQEG* describes in table 1 how to determine the Quasi Essential Graph Q associated to a set of DAGs $\mathcal{B}$ and a given threshold α.

In order to avoid problems related to Markov equivalence, the preprocessing phase consists in considering essential graphs corresponding to elements of $\mathcal{B}$ and in estimating frequency of each undirected edge (steps 3 to 6).

This algorithm then comprises two phases: we first determine during steps 8 to 10 the skeleton of Q and estimate the weights w_u of each undirected edge using condition C_1 of QEGs (cf. section 3.1). We then estimate during steps 11 to 19 the potential weight of each edge orientation in the set of DAGs $\mathcal{B}$ and use condition C_2 of QEGs to possibly keep consistent arrows and their corresponding weights w_a.

3.4 Visualization

Inspiration of this work is related to the fact that visualizing a set of DAGs is not very practical. Quasi Essential Graph has been defined as a graphical object. We propose here one graphical metaphor for visualizing it.

For classical weighted graph, weights are only defined for edges, and the graphical metaphor is simply obtained by varying the size of the corresponding line. As QEG associates weights to both undirected edges and arrows, we will then vary the size of the corresponding lines and arrows, and use different colors for well differentiating both graphical items.

3.5 Illustration with a toy axample

As this work is a preliminary work, we propose here a toy example illustrating the interest of our QEG approach. Our various algorithms have been implemented in C++ with the Boost library (`http://www.boost.org`) and APIs provided by the ProBT platform (`http://bayesian-programming.org`). Visualization tools are provided by Tulip framework (`http://tulip.labri.fr`).

Figure 1 gives us a set of 12 DAGs generated by randomly varying the first DAG. These variations involve adding, removing or inverting between one and three edges in the initial DAG.

Figure 2(a) describes the QEG skeleton obtained after the first phase of our algorithm (with $\alpha = 0.55$). We can see that it returns here the skeleton of the initial and perturbed DAG (DAG1). Figure 2(b) shows us the consistent arrows added during the second phase of our algorithm. We can notice here that this graph corresponds to the Essential Graph of DAG1. This result is coherent with the QEG definition.

Figure 2(c) proposes the final QEG using our graphical metaphor. This figure confirms that QEG is a practical way to represent a set of DAGs, providing in one unique graph a consistent summary of all these graphs.

4 Conclusion

We propose in this work the notion of Quasi Essential Graph (QEG) in order to summarize a set of Bayesian networks. We detailed here some interesting

Algorithm $DAGsToQEG(\mathcal{B}, \alpha)$

Require: A set of DAGs, $\mathcal{B}$, and a threshold, α ($\alpha > 0.5$).
Ensure: A unique Quasi Essential Graph Q.

1: $N \leftarrow card(\mathcal{B})$;
2: $Q \leftarrow \emptyset$;
3: **for** $i = 1$ to N **do**
4: $\quad EG_i \leftarrow EG(\mathcal{B}(i))$; {Preprocessing Phase}
5: **end for**
6: $[UG, w(u)] \leftarrow Union\{Skeleton(EG_i)\}$;
7: **for** every edge $u \in UG$ **do**
8: $\quad$ **if** $w(u) > \alpha$ **then**
9: $\quad\quad add\_edge(u, Q)$; {Skeleton determination}
10: $\quad\quad w_u(u) = w(u)$;
11: $\quad\quad w(\overleftarrow{u}) = card(\{EG_i | \overleftarrow{u} \in EG_i\})/(N * w(u))$; {Arrows determination}
12: $\quad\quad w(\overrightarrow{u}) = card(\{EG_i | \overrightarrow{u} \in EG_i\})/(N * w(u))$;
13: $\quad\quad$ **if** $w(\overleftarrow{u}) > \alpha$ **then**
14: $\quad\quad\quad orient\_edge(\overleftarrow{u}, Q)$;
15: $\quad\quad\quad w_a(\overleftarrow{u}) \leftarrow w(\overleftarrow{u})$;
16: $\quad\quad$ **else if** $w(\overrightarrow{u}) > \alpha$ **then**
17: $\quad\quad\quad orient\_edge(\overrightarrow{u}, Q)$;
18: $\quad\quad\quad w_a(\overrightarrow{u}) \leftarrow w(\overrightarrow{u})$;
19: $\quad\quad$ **end if**
20: $\quad$ **end if**
21: **end for**
22: **return** Q

Notations :
- $EG(\mathcal{B}(i))$: algorithm returning the Essential Graph of a DAG (cf. sect.2.2)
- $Union\{Skeleton(EG_i)\}$: algorithm generating an undirected graph resulting to the union of the set of skeletons. This algorithm counts the frequency of each undirected edge in this set
- for an undirected edge $u = X_i$–X_j, the two possible orientations are $\overleftarrow{u} = X_i \leftarrow X_j$ and $\overrightarrow{u} = X_i \rightarrow X_j$

Table 1. Algorithm returning the Quasi Essential Graph Q for a set of DAGs $\mathcal{B}$ and for a given threshold α

properties with respect to the BNs we want to summarize. We then described an algorithm to determine the QEG from a set of BNs and a given threshold.

One main advantage of this QEG, and inspiration of this work, is that this new object is easily visualized with a graphical metaphor we proposed here and illustrated on a toy example.

Both theoretical properties and experimental results have now to be extended. From a theoretical point of view, we want to go further about parametrization of a DAG distribution. More realistic experiments are also

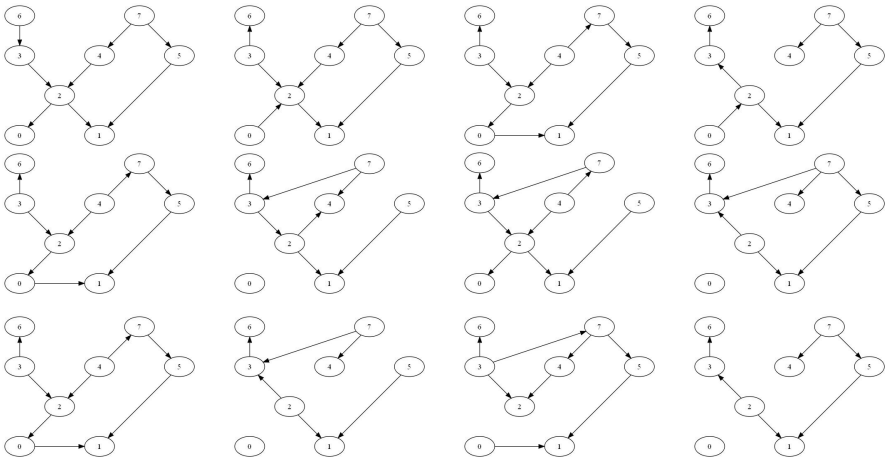

Fig. 1. Set of 12 DAGs obtained by randomly perturbing the first one.

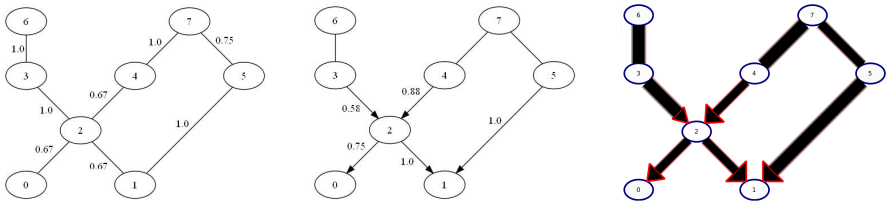

Fig. 2. (a) QEG skeleton obtained after the first phase of our algorithm (b) consistent arrows added to the previous skeleton after the second phase of our algorithm (c) QEG graphical metaphor using Tulip framework

to be performed with sets of BNs obtained from ensemble structure learning such as evolutionary algorithms or bootstrap approaches.

Acknowledgment

This work was partially supported by a grant from the Pays de la Loire Region, Bioinformatics Research Project (BIL).

References

[AdF07]C. Auliac, F. dAlchéBuc, and V. Frouin. Learning transcriptional regulatory networks with evolutionary algorithms enhanced with niching. In F. Masulli, S. Mitra, and G. Pasi, editors, *Applications of Fuzzy Sets Theory*, volume 4578 of *Lecture Notes in Computer Science*, pages 612–619. Springer Berlin / Heidelberg, 2007.

[AMP95]S. Andersson, D. Madigan, and M. Perlman. A characterization of markov equivalence classes for acyclic digraphs. Technical Report 287, Department of Statistics, University of Washington, 1995.

[CGH94]D. Chickering, D. Geiger, and D.E. Heckerman. Learning bayesian networks is NP-hard. Technical Report MSR-TR-94-17, Microsoft Research, 1994.

[CH96]D. Chickering and D. Heckerman. Efficient Approximation for the Marginal Likelihood of Incomplete Data given a Bayesian Network. In *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence (UAI-96)*, pages 158–168. Morgan Kaufmann, 1996.

[Chi02]D. Chickering. Learning equivalence classes of Bayesian-network structures. *Journal of Machine Learning Research*, 2:445–498, February 2002.

[DBC07]A. Delaplace, T. Brouard, and H. Cardot. Two evolutionary methods for learning bayesian network structures. In Y. Wang, Y. Cheung, and H. Liu, editors, *Computational Intelligence and Security*, volume 4456 of *Lecture Notes in Computer Science*, pages 288–297. Springer Berlin / Heidelberg, 2007.

[DT92]D. Dor and M. Tarsi. A simple algorithm to construct a consistent extension of a partially oriented graph. Technical Report R-185, Cognitive Systems Laboratory, UCLA Computer Science Department, 1992.

[FGW99]N. Friedman, M. Goldszmidt, and A. J. Wyner. Data analysis with bayesian networks: A bootstrap approach. In *Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI–99)*, pages 206–215, San Francisco, CA, 1999. Morgan Kaufmann Publishers.

[LPY+96]Larranaga, M. Poza, Y. Yurramendi, R. Murga, and C. Kuijpers. Structure learning of bayesian networks by genetic algorithms: A performance analysis of control parameters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):912–926, 1996.

[MC07]J. Muruzábal and C. Cotta. A study on the evolution of bayesian network graph structures. In P. Lucas, J. Gàmez, and A. Salmerón, editors, *Advances in Probabilistic Graphical Models*, volume 214 of *Studies in Fuzziness and Soft Computing*, pages 193–213. Springer Berlin / Heidelberg, 2007.

[Nea03]R. E. Neapolitan. *Learning Bayesian Networks*. Prentice Hall, 2003.

[RB05]A. S. Rodin and E. Boerwinkle. Mining genetic epidemiology data with Bayesian networks I: Bayesian networks and example application (plasma apoE levels). *Bioinformatics*, 21(15):3273–3278, 2005.

[SM95]P. Spirtes and C. Meek. Learning bayesian networks with discrete variables from data. In *Proceedings of First International Conference on Knowledge Discovery and Data Mining*, pages 294–299. AAAI Press, 1995.

[VP91]T. Verma and J. Pearl. Equivalence and synthesis of causal models. In M. Henrion, R. Shachter, L. Kanal, and J. Lemmer, editors, *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, pages 220–227, San Francisco, 1991. Morgan Kaufmann.

[WY10]T. Wang and J. Yang. A heuristic method for learning bayesian networks using discrete particle swarm optimization. *Knowledge and Information Systems*, 24:269–281, 2010.