Bayesian Network Exploratory Tool

Elad Cohen

April 7, 2017

1 Introduction

Probabilistic Graphical Models (PGM) are useful constructs for describing complex probability distributions over high dimensional space. Bayesian networks are a common type of PGMs described using a directed graph, in which nodes correspond to random variables and edges correspond to direct dependencies between variables. The power of PGMs comes from the fact that they factor a large complex joint probability distribution into a product of much smaller probability tables. The graph structure allows for an intuitive visual representation of the network of dependencies between variables. Additionally, formulation of a high-dimensional probability distribution using PGMs enables efficient algorithms for inference from data, and even learning the underlying probability.
The main objective of this project is to develop a user-friendly software tool for constructing Bayesian networks that allows interactive exploration of different aspects and inference algorithms. This tool will be designed both for educational use in academic courses and for self exploration by scientists and developers in the industry. The proposed software tool will contain six main functional units (see Section 2 for more detailed description):

1. Unit 1 - conditional independence in Bayesian networks (D separation criterion).
2. Unit 2 - undirected representations of Bayesian networks and the concept of triangular (or chordal) graphs.
3. Unit 3 - elimination orders. Examine the influence of elimination order on the complexity of the underlying inference algorithm.
4. Unit 4 - inference algorithms. Inferring conditional distributions given observed data using elimination and message passing.
5. Unit 5 - parameter inference.
6. Unit 6 - sampling algorithms on Bayesian networks (MCMC, loopy belief propagation).

There are several existing software tools for PGMs:

1. Bayesian DAG learning [1] - Bayesian inference about directed acyclic graph (DAG) structures using dynamic programming and MCMC (Markov Chain Monte Carlo). This tool is free and it's written in Matlab. The main objective is to facilitate computation, designed to run on small amount of variables (up to 20). It's main features are:

(a) Computes exact edge marginals using Bayesian model averaging with dynamic programming

(b) Computes edge marginals approximately using MCMC

2. Probabilistic Graphical Models [4] - Inference of Bayesian networks and Markov chains. This tool includes learning examples as well. Implemented in Matlab with following features:

   (a) Joint distribution calculation

   (b) Marginal distribution calculation

   (c) Maximum probability assignment calculation

   (d) Inference on tree structured network

   Other relevant tools, e.g OpenGM [3] (Discrete factor graph models), Hidden Markov Model Toolbox [2] (Hidden Markov Model), MAP estimation of DAG structure [5], are more specialized and do not contain the main features of our proposed tool.

Existing libraries have some aspects in common with our tool, but none of them appear to have the interactive nature that will support friendly exploration and learning. Additionally, none of these tools possess the gradual unit structure we propose, which we believe will help convey complex concepts in statistical learning.

2 Proposed Design

The software tool will consist of six sequential functional units. A Bayesian network is represented via a directed acyclic graph, where each node represents a random variable, with conditional distribution table given its parents (or predecessors) in the graph. The visual representation of the graph allows users to explore the network and observe the different conditional distribution tables.

1. Unit 1 – fundamental concept of conditional independence in Bayesian networks. This unit considers only the graph structure of the network and ignores the probability tables.

   (a) Marginal independence. The user selects two nodes in the network, and the program specifies whether they are independent or not. If they are dependent, then the connecting path is highlighted.

   (b) Conditional independence. The user selects two nodes u, v and a set of nodes to condition on A, and the program specifies whether X_u and X_v are conditionally independent given the collection X_A ($X_V \perp\!\!\!\perp X_U | X_A$). We will use the D-separation criterion for this and highlight the path that blocks dependence.

2. Unit 2 – undirected graph representation of Bayesian networks. This unit also considers only the graph structure of the network and ignores the probability tables.

(a) Convert the directed graph representation to an undirected graph, called the moralized graph. In this undirected representation parents of a node in the original graph will have a new edge between them. Consequently, all variables in the same conditional probability table form a clique.

(b) Examine whether the moralized graph is chordal (triangulated) or not. In a chordal graph, every cycle of four or more vertices has a chord. Note that a perfect elimination exists, iff the moralized graph is chordal (see Unit 3).

(c) Convert the moralized graph into a factor graph – a bipartite graph representation with one class representing nodes in original graph and the other class representing maximal cliques (probability distribution tables).

3. Unit 3 – Elimination algorithm on undirected graph. Helps choose an elimination order for the purpose of inference without carrying out the complex computations of the probability tables.

   (a) Show elimination algorithm implementation on graphs. User can run elimination algorithm step by step. On every step the user choose a node to eliminate, on the graph the user can see the new edges added to the graph.

   (b) Try to find a prefect elimination order, choose one simplical node and start elimination from there, write the order on the screen, on every step find the next simplical node, if there is not such one, the user will be able to choose the next node to eliminate.

4. Unit 4 – Inference of hidden variables using observed variables. In this unit the main objective is to show calculation of joint, marginal and max probability of random variables. Show inference on graph representation (undirected, factor) of Bayesian network along with probability tables.
   As first step the user selects the observed nodes on the network, for each one sets the node value (for elimination algorithm choose the nodes to inference).
   Propose to the user three modes of inference:

   (a) Simple elimination - find marginal distribution for a set of variables given another set of observed variables.

   (b) Message Passing - compute marginal distribution for several variables separately given a set of observed variables.

   (c) Parallel Message Passing - same computation as in the last item but run the procedure in parallel (cannot run the step by step execution mode)

   Each one of the inference methods mentioned above can be run in one of the execution modes:

   (a) Full run, in this mode the algorithm runs the calculations in the background. When the results are ready, the user is able to see marginal for a node by selecting that node.

   (b) Step by step, on every step do a single algorithm calculation.

   Message passing algorithm is a special case were we can do different calculations on the graph representation. And it applies to both simple and parallel message passing. The user can have the ability to run the algorithm to calculate:

(a) Marginal distribution - calculates marginal distribution for every node

(b) Max probability assignment - computes assignment of unobserved nodes with observed nodes

5. Unit 5 – Parameter inference using observations. The networks this unit is operating on are HMM (Hidden Markov Model) and small Bayesian network with many independent copies.

   The user has the ability to choose a couple of modes:

   (a) Inference where all nodes are observed. is data to sufficient statistics table and display it to the user.

   (b) Inference where some of the nodes are observed and some are hidden using:

      i. Maximum probability inference

      ii. EM (expectation maximization)

6. Unit 6 – Other advanced algoriths such as:

   (a) Markov chain Monte Carlo algorithm

   (b) Loopy Belief Propagation

3 Implementation Demo

For a demonstration of the tool, we constructed a network in the tool with 5 binary random variables, and added dependencies between them. Figure 1 demonstrate unit 1, the network as DAG in the tool. Since X_4 was clicked, it is highlighted and the conditional table $P(X_4|X_3)$ is shown. Figure 2 shows the binary membership table. This table show a view of all nodes participant in probability function, and in addition shows all the probability functions a node participant in. Figure 3 shows an undirected version of the network with the binary membership table, which demonstrate unit 2.
Elimination of a node X_4 is demonstrated in 4, we can see how one node is eliminated and there are new edges introduced in the undirected graph representation of the network. This demonstration for unit 3 in our tool.

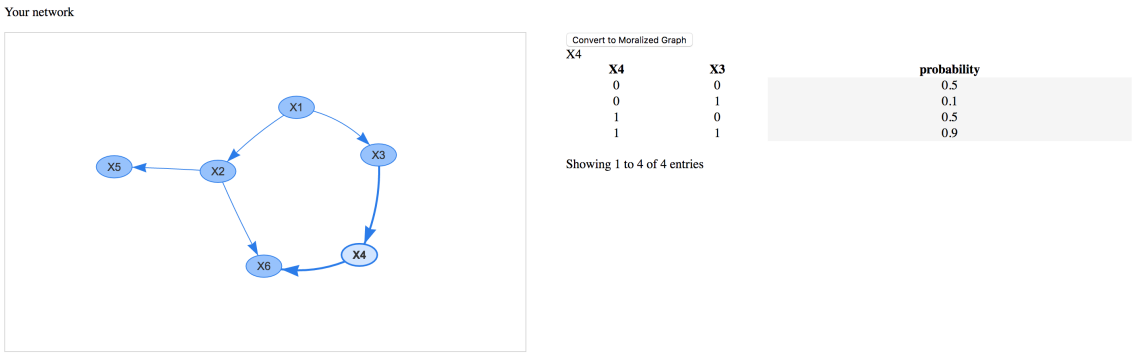

Figure 1: A simple Bayesian network representation using directed graph

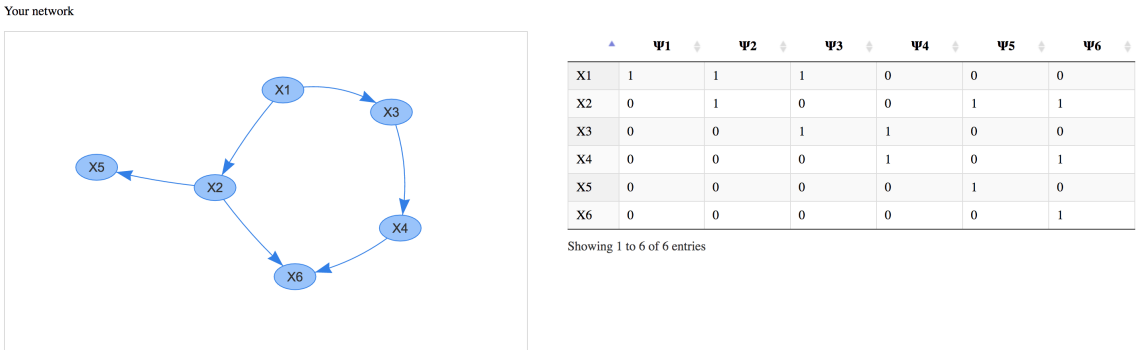

Figure 2: Showing binary membership table of the network

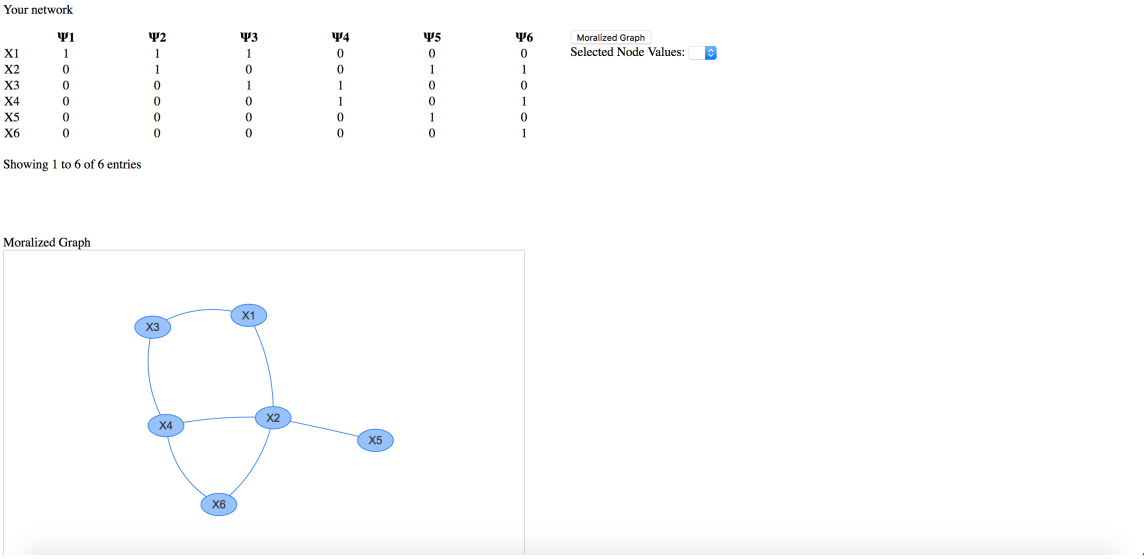

Figure 3: Moralized Graph representation

References

[1] Kevin Murphy, PGM Matlab libraries,
`http://www.cs.ubc.ca/~murphyk/Software/`

[2] Kevin Murphy, toolbox for inference on Hidden Markov Models,
`http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html`

[3] Bjoern Andre, Thorsten Beier and Joerg H. Kappes, OpenGM,
`http://hciweb2.iwr.uni-heidelberg.de/opengm/`

[4] Syllogismos, PGM learning,
`https://github.com/anhncs/Probabilistic-Graphical-Models`

Your network

	Ψ1	Ψ2	Ψ3	Ψ5
X1	1	1	1	0
X2	0	1	1	0
X3	0	0	1	0
X5	0	0	0	0

Showing 1 to 4 of 4 entries

Moralized Graph

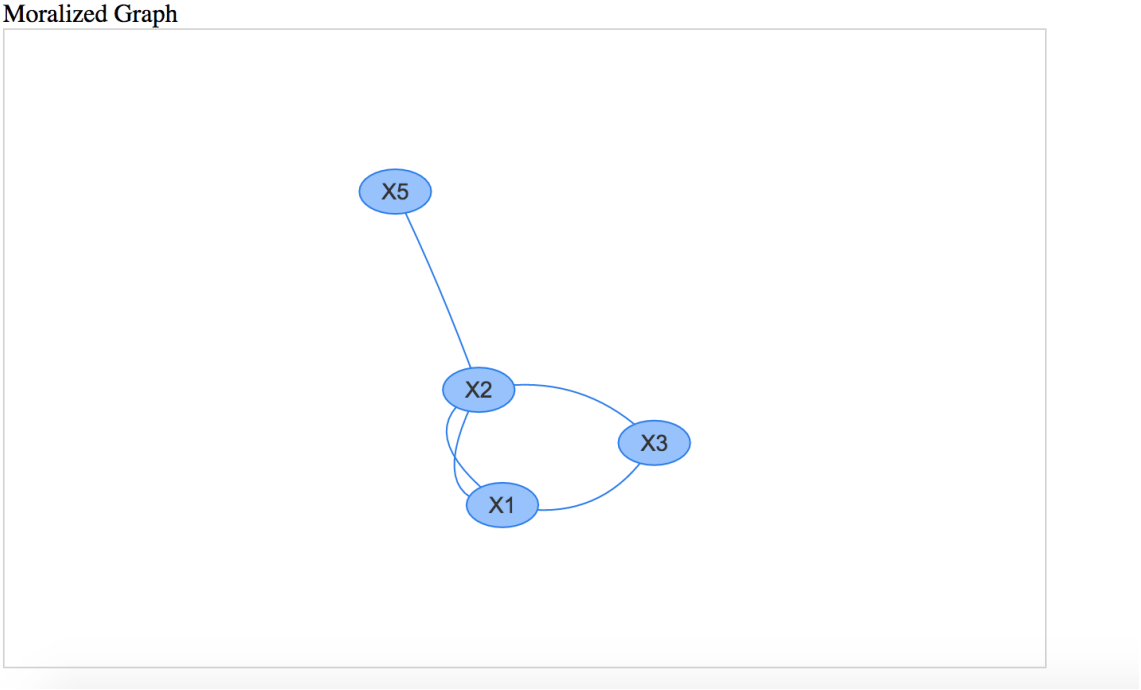

Figure 4: After X4 is eliminated from the graph. We can see a new edge was created between X2 and X3. Now X5 node is chosen to be eliminated next

[5] Mark Schmidt and Kevin Murphy, MAP estimation of DAG structures, `http://www.cs.ubc.ca/ murphyk/Software/DAGlearn/index.html`