Homework 1: Belief Propagation & Factor Graphs

Brown University CS 242: Probabilistic Graphical Models

Homework due at 11:59pm on October 5, 2016

We examine the problem of computing marginal distributions in graphical models, possibly given observations of the variables at some nodes. We will use factor graph representations of the target models, and implement the sum-product variant of the belief propagation algorithm to compute marginals. To understand the details of the sum-product algorithm, we recommend Chapter 5 of Barber's "Bayesian Reasoning and Machine Learning", as well as "Factor Graphs and the Sum-Product Algorithm" by Kschischang, Frey, & Loeliger, *IEEE Trans. Information Theory* 47, pp. 498-519, 2001.
You must write your own novel implementation of the sum-product algorithm, not copy code from other students or existing software packages.

Question 1:

We have provided Matlab code implementing a data structure to store the graph adjacency structure, and numeric potential tables, defining any discrete factor graph. We also provide code that explicitly builds a table containing the probabilities of all joint configurations of the variables in a factor graph, and sums these probabilities to compute the marginal distribution of each variable. Such "brute force" inference code is of course inefficient, and will only be computationally tractable for small models.

We recommend (but do not require) that you use these same data structures for your own implementation of the sum-product algorithm, by implementing `run_loopy_bp_parallel.m` and `get_beliefs.m`. To gain intuition for the graph structure, examine the output of `make_debug_graph.m`. Think of the code we provide as a starting point: you are welcome to create additional functions or data structures as needed. You may use other programming languages if you wish, but will be responsible for translating the Matlab-compatible data we provide into other formats.

a) *Implement the sum-product algorithm. Your code should support an arbitrary factor graph linking a collection of discrete random variables. Use a parallel message update schedule, in which all factor-to-variable messages are updated given the current variable-to-factor messages, and then all variable-to-factor messages given the current factor-to-variable messages. Initialize by setting the variable-to-factor messages to equal 1 for all states. Be careful to normalize messages to avoid numerical underflow.*
Hints: While it is acceptable to make significant use of loops in your sum-product code, other implementation strategies will lead to faster experiments. In Matlab, a standard

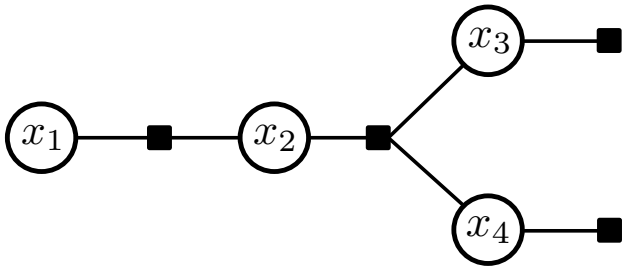

Figure 1: A tree-structured factor graph in which four factors link four random variables. Variable x_2 takes one of three discrete states, and the other three variables are binary.

vector is a multi-dimensional array in which the first dimension has size equal to the vector's length, and all other dimensions have length one. The `reshape` command can convert vectors to arrays where some other dimension (chosen to match a factor array) has length greater than one. To easily compute the product of a message array and a factor array, apply `bsxfun` to the `@times` function. The `sum` command provides an optional dimension argument, to allow marginalization of any dimension in an array.

b) *Consider the four-node, tree-structured factor graph illustrated in Figure 1. Variable x_2 takes one of three discrete states, and the other three variables are binary. Numeric values for the potential functions are defined in* `make_debug_graph.m`*. Run your implementation of the sum-product algorithm on this graph, and report the marginal distributions it computes. Verify that these are consistent with the results of* `marg_brute_force.m`*.*

Question 2:

We apply our sum-product inference code to the *ALARM: A Logical Alarm Reduction Mechanism* network, an early example of an expert system designed for hospital patient monitoring. Figure 2 illustrates this network as a directed graphical model.

To allow your sum-product implementation to be easily applied to this model, the Matlab function `make_alarm_graph.m` constructs a factor graph representation of the ALARM network, in which one factor node links each variable to its parents. The function `add_evidence.m` modifies a factor graph to account for observations by taking appropriate "slices" of the original factors; use this to condition on data. In some of the questions below, we consider smaller networks containing a subset of the nodes in the full ALARM network. Remember that in a directed graphical model, an unobserved variable with no children can be exactly marginalized by simply removing its associated factor and variable nodes from the graph.

All variables in the ALARM network are discrete, and take 2 to 4 states indexed by positive integers. For some questions below, we ask you to report means of certain marginal distributions. Means are sensible for the discrete variables in this model, because all variables are defined on an ordinal scale in which larger values correspond to greater degrees of presence or severity of a measurement or diagnosis.

For each part below, run the sum-product algorithm until the maximum change in message values between iterations (an update of all messages) is at most 10^{-6}. If sum-product fails to converge after 500 iterations, report this and discuss possible explanations.

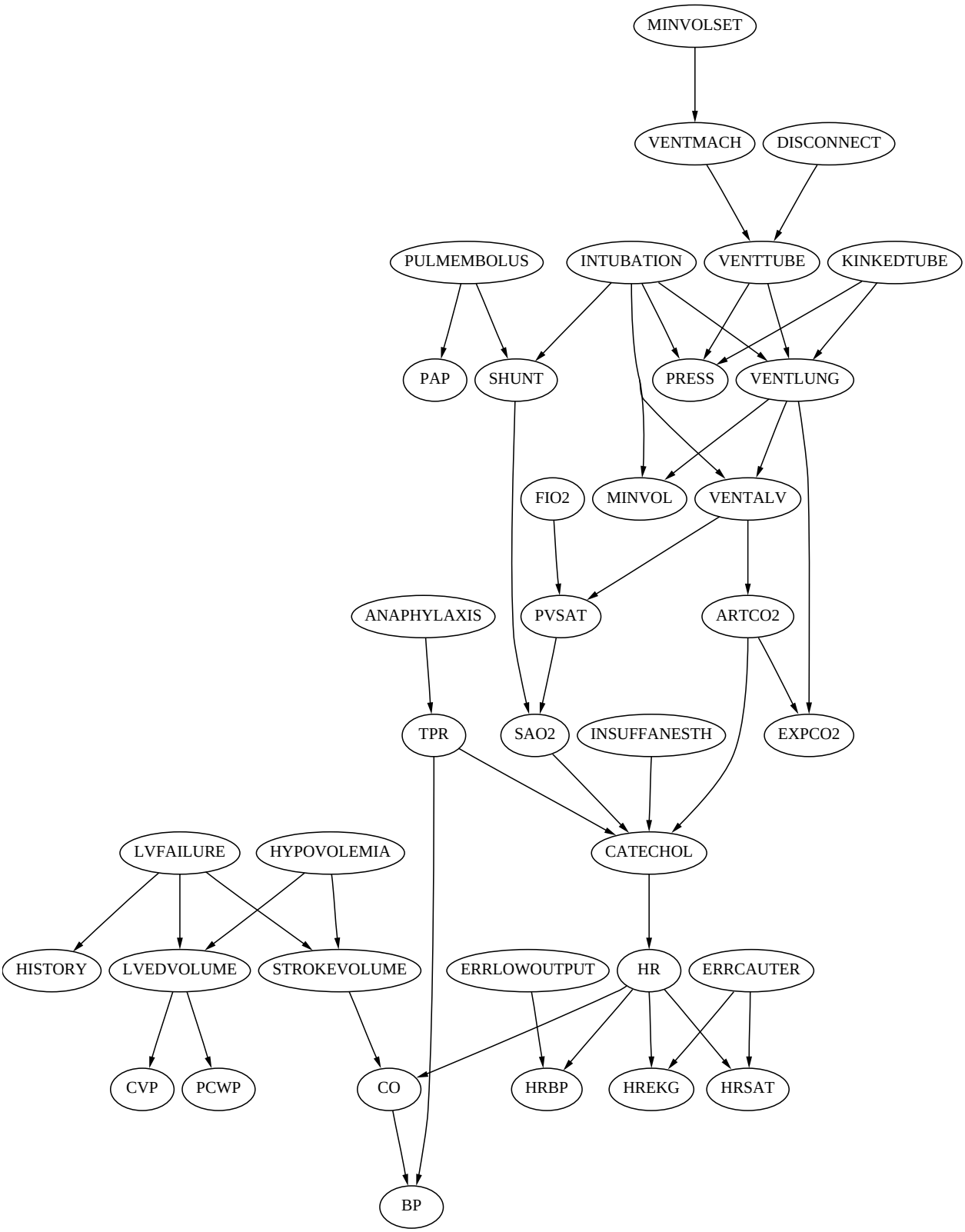

Figure 2: The ALARM network represented as a directed graphical model. To apply the sum-product algorithm, we create a factor graph in which one factor links each variable to its parents.

a) *Consider a sub-network containing four "cause" variables (*PULMEMBOLUS, INTUBATION, VENTTUBE, KINKEDTUBE*) and four "effect" variables (*PAP, SHUNT, PRESS, VENTLUNG*). To construct this graph, we condition on observing* VENTMACH=4, DISCONNECT=2, *and (exactly) marginalize all other descendants; an implementation is in* make_alarm_graph_partA.m. *Use sum-product to compute the means of the four cause variables, and compare your answer to that produced by* marg_brute_force.m.

b) *Take the network from part (a), and also condition on observing* SHUNT=2, PRESS=4. *Use sum-product to compute the means of the four cause variables, and compare your answer to that produced by exact enumeration-based inference.*

c) *Now consider a different sub-network containing the variable nodes* INTUBATION, VENTTUBE, KINKEDTUBE, *and* VENTLUNG. *To construct this graph, condition on observing* VENTMACH=4, DISCONNECT=2, PRESS=4, MINVOL=2, *and (exactly) marginalize all other descendants; an implementation is in* make_alarm_graph_partC.m. *Use sum-product to compute the means of the four unobserved nodes, and compare your answer to that produced by exact enumeration-based inference.*

d) *Discuss the results in parts (a-c), explaining cases in which sum-product produces exact marginals, and cases in which they only approximate the true marginals.*

e) *Now consider the full ALARM network, and condition* HYPOVOLEMIA, HR, INTUBATION, KINKEDTUBE, *and* VENTALV *to take their smallest possible values. Use the sum-product algorithm to compute the means of the diagnostic variables* LVFAILURE, ANAPHYLAXIS, INSUFFANESTH, PULMEMBOLUS, *and* DISCONNECT. *Do you expect sum-product to compute exact marginal distributions for this graph? Why or why not?*

f) *Consider the same scenario as in part (e), and suppose we also observe that* HRBP, HREKG, *and* HRSAT *take their largest possible values. How do the means of the diagnostic variables change? Briefly explain what you observe.*

g) *Consider the full ALARM network with no observations. Using the sum-product algorithm, compute the means of these eight diagnostic variables:* LVFAILURE, HYPOVOLEMIA, ANAPHYLAXIS, INSUFFANESTH, PLUMEMBOLUS, INTUBATION, DISCONNECT, KINKEDTUBE. *Are these estimates equal to the true means?*

h) *Consider the full ALARM network, and condition the following observation variables to their smallest possible values:* HISTORY, CVP, PCWP, BP, HRBP, HREKG, HRSAT, EXPCO2, MINVOL. *Use the sum-product algorithm to compute the means of the same 8 diagnostic variables, and discuss the most significant changes.*

i) *Consider the network and observations from part (h), but change* HRBP, HREKG, *and* HRSAT *to take their largest possible values. Use the sum-product algorithm to compute the means of the same 8 diagnostic variables, and discuss the most significant changes.*

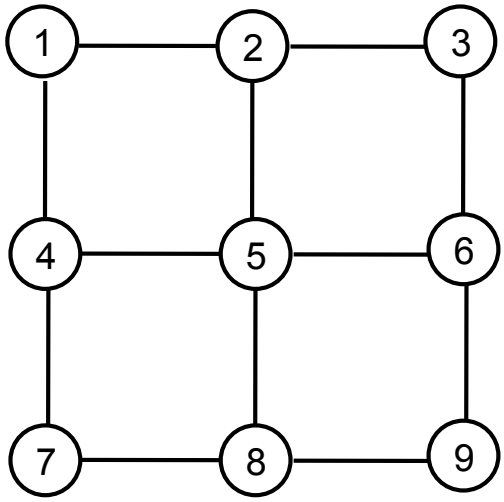

Figure 3: An undirected graphical model of two-dimensional spatial dependencies among nine variables arranged in a 3×3 grid. All variables are discrete, and have the same number of states.

Question 3:

For a graph with cycles like the ALARM network, the sum-product algorithm may not always compute the correct marginal distributions. Here we instead explore the computational complexity of the elimination algorithm, which is guaranteed to find exact marginals with additional computational expense. Consider the undirected graphical model in Figure 3, which is a small example of the spatial lattices used in computer vision applications.

a) Consider the elimination algorithm discussed in lecture, and in Sec. 5.3.1 of Barber's textbook. Suppose we compute the marginal of node 1 via the following elimination ordering: $\{5, 2, 6, 8, 4, 9, 7, 3, 1\}$. *Sketch the sequence of graphs formed. What is the largest clique that is produced?*

b) Suppose we instead compute the marginal of node 1 via the following elimination ordering: $\{9, 7, 3, 6, 8, 5, 2, 4, 1\}$. *Sketch the sequence of graphs formed. What is the largest clique that is produced? Which ordering is more computationally efficient?*

c) Using intuition from these examples, give an upper bound on the treewidth of an $n \times n$ *grid. Explain your answer, which should be much smaller than the number of nodes* n^2.