Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science

6.438 Algorithms For Inference
Fall 2014

2 Directed Graphical Models

Today we develop the first class of graphical models in the course: directed graphical models. A directed graphical model defines a *family* of joint probability distributions over a set of random variables. For example, suppose we are told that two random variables x and y are independent. This characterizes a family of joint distributions which all satisfy $p_{\mathsf{x},\mathsf{y}}(x, y) = p_{\mathsf{x}}(x)p_{\mathsf{y}}(y)$. Directed graphs define families of probability distributions through similar factorization properties.

A directed graphical model $\mathcal{G}$ consists of nodes $\mathcal{V}$ (representing random variables) and directed edges (arrows) $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. (The notation $(i, j) \in \mathcal{E}$ means that there is a directed edge from i to j.)

Directed graphs define families of distributions which factor by functions of nodes and their parents. In particular, we assign to each node i a random variable x_i and a non-negative-valued function $f_i(x_i, x_{\pi_i})$ such that

$$\sum_{x_i \in \mathcal{X}} f_i(x_i, x_{\pi_i}) = 1, \tag{1}$$

$$\prod_i f_i(x_i, x_{\pi_i}) = p(x_1, \ldots, x_n), \tag{2}$$

where π_i denotes the set of parents of node i. Assuming the graph is *acyclic* (has no directed cycles), we must have $f_i(x_i, x_{\pi_i}) = p_{\mathsf{x}_i|\mathsf{x}_{\pi_i}}(x_i|x_{\pi_i})$, i.e., $f_i(\cdot, \cdot)$ represents the conditional probability distribution of x_i conditioned on its parents. If the graph does have a cycle (e.g. Figure 1), then there is no consistent way to assign conditional probability distributions for the cycle. Therefore, we assume that all directed graphical models are directed acyclic graphs (DAGs).

In general, by the chain rule, any joint distribution of any n random variables $(\mathsf{x}_1, \ldots, \mathsf{x}_n)$ can be written as

$$p_{\mathsf{x}_1,\ldots,\mathsf{x}_n}(x_1, \ldots, x_n) = p_{\mathsf{x}_1}(x_1)p_{\mathsf{x}_2|\mathsf{x}_1}(x_2|x_1) \cdots p_{\mathsf{x}_n|\mathsf{x}_1,\ldots,\mathsf{x}_{n-1}}(x_n|x_1, \ldots, x_{n-1}). \tag{3}$$

By treating each of these terms as one of the functions f, we observe that the distribution obeys the graph structure shown in Figure 2. This shows that DAGs are *universal*, in the sense that any distribution can be represented by a DAG. Of particular interest are *sparse* graphs, i.e. graphs where the number of edges is much smaller than the number of pairs of random variables. Such graphs can lead to efficient inference.

In general, the graph structure plays a key role of determining the size of the representation. For instance, we saw above that a fully connected DAG can represent an arbitrary distribution, and we saw in Lecture 1 that the joint probability table for

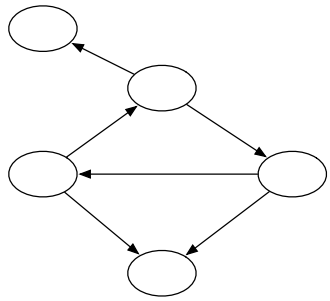


Figure 1: Example of a directed graph with a cycle, which does not correspond to a consistent distribution.

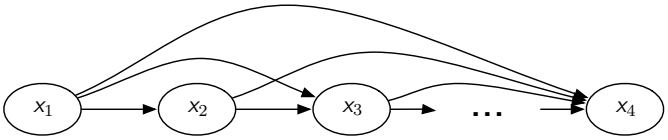


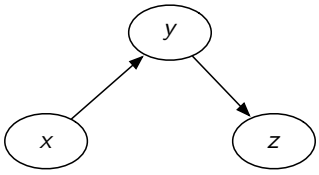
Figure 2: A fully connected DAG is universal, in that it can represent any distribution.

such a distribution requires $|\mathcal{X}|^N$ entries. More generally, the number of parameters required to represent the factorization is of order $|\mathcal{X}|^{\max_i |\pi_i|}$, which is dramatically smaller if $\max_i |\pi_i| \ll N$. Similarly, the graph structure affects the complexity of inference: while inference in a fully connected graph always requires $|\mathcal{X}|^N$ time, inference in a sparse graph is often (but not always) much more efficient.

There is a close relationship between conditional independence and factorization of the distribution. We'll first analyze at some particular examples from first principles, then look at a more general theory.

2.1 Examples

Example 1. First, consider the following graph:



This graph represents the factorization

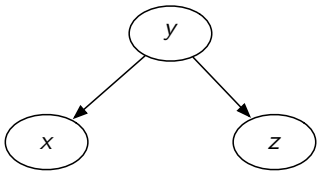
$$p_{x,y,z} = p_{z|y} p_{y|x} p_x.$$

By matching these terms against the chain rule for general probability distributions

$$p_{x,y,z} = p_{z|y,x} p_{y|x} p_x,$$

we see that $p_{z|y} = p_{z|y,x}$, i.e. $x \perp\!\!\!\perp z|y$.

Example 2. Now consider the graph:

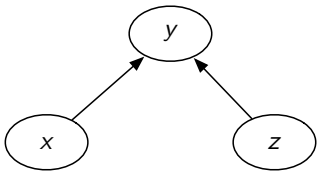


This graph represents the factorization

$$P_{x,y,z} = p_{z|y} p_{x|y} p_y.$$

We can match terms similarly to the above example to find that $x \perp\!\!\!\perp z|y$ in this graph as well.

Example 3.

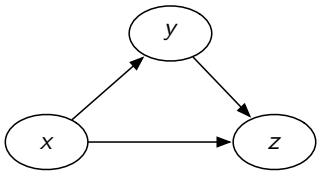


The factorization is:

$$p_{x,y,z} = p_x p_{y|x,z} p_z.$$

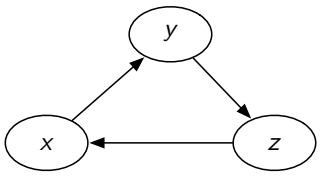
By matching terms, we find that $x \perp\!\!\!\perp z$. However, it's no longer true that $x \perp\!\!\!\perp z|y$. Therefore, we see that the direction of the edges matters. The phenomenon captured by this example is known as *explaining away*. (Suppose we've observed an event which may result from one of two causes. If we then observe one of the causes, this makes the other one less likely – i.e., it explains it away.) The graph structure is called a *v-structure*.

Example 4.

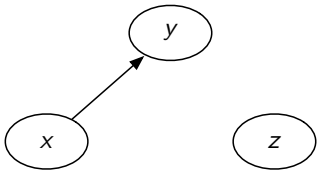


This is a fully connected graph, so as we saw before, it can represent *any* distribution over x, y, and z. That is, we must remove edges in order to introduce indepence structure.

Example 5. The following graph structure is not a valid DAG because it contains a cycle:



Example 6. The following graph is obtained by removing an edge from Example 3:

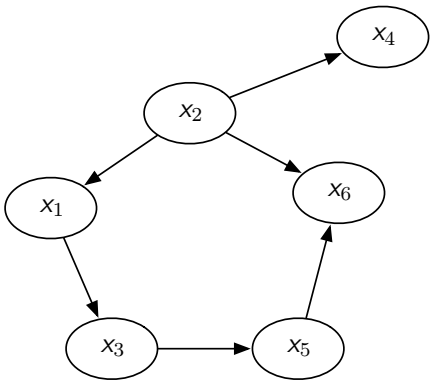


The factorization represented by this graph is

$$p_{x,y,z} = p_x p_{y|x} p_z,$$

a strict subset of the factorization in Example 3. In general, reducing edges increases the number of independencies and decreases the number of distributions the graph can represent.

Example 7. Here is a bigger example with more conditional independencies.



As before, we can identify many of them using the factorization properties. In particular, we can read off some conditional independencies by doing the following:

- Choose a *topological ordering* of the nodes (i.e. an ordering where any node i comes after all of its parents).
- Let ν_i be the set of nodes that are not parents of i, i.e. $\pi_i \cap \nu_i = \emptyset$, but they appear in the topological ordering before i.
- Then the graph implies the conditional independence $x_i \perp\!\!\!\perp x_{\nu_i} | x_{\pi_i}$.

Note that there may be many topological orderings for a graph. With the above procedure, different conditional independences can be found using different topological orderings. Now, we discuss a simpler and more general procedure for testing conditional independence which does not depend on any particular topological ordering.

2.2 Graph Separation and Conditional Independence

We now introduce the idea of graph separation: testing conditional independence properties by looking for particular kinds of paths in a graph. From Examples 1 and 2 above, we might be tempted to conclude that two variables are dependent if and only if they're connected by a path which isn't "blocked" by an observed node. However, this criterion fails for Example 3, where x and z are dependent only when the node between them *is* observed. We can repair this broken intuition, however, by defining a different set of rules for when a path is blocked.

2.2.1 d-separation and Bayes Ball

Let A, B, and C be disjoint subsets of the set of vertices $\mathcal{V}$. To test whether x_A and x_B are conditionally independent given x_C:

1. Shade nodes in C. Call this *primary* shade of a node. In addition, assign *secondary* shade to each node as follows:
   - All nodes with *primary* shade also have *secondary* shade.
   - All nodes that are parent of a node with *secondary* shade have a *secondary* shade.
2. Place a ball at each node in A.
3. Let balls bounce around in the graph following rules shown in Figure 3.

   Remark: Balls do not interact. The shade in rules 1 and 2 is from *primary* shading only. While in rule 3, both *primary* and *secondary* shading applies.
4. If no ball can reach any node in B, then x_A must be conditionally independent of x_B given x_C.

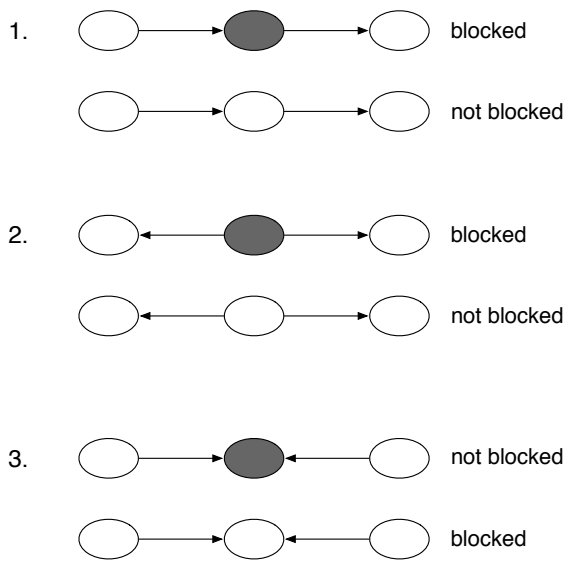


Figure 3: Rules for blocking and non-blocking in the Bayes Ball algorithm.

2.3 Characterization of DAG's

The following two characterizations are equivalent descriptions of probability distributions:

1. Factorization into a product of conditional probability tables according to the DAG structure
2. Complete list of conditional independencies obtainable by Bayes Ball

Another way of stating this is that the following two lists are equivalent:

1. List all distributions which factorize according to the graph structure.
2. List all possible distributions, and list all the conditional independencies obtainable by Bayes ball. Discard the distributions which do not satisfy all the conditional independencies.

2.4 Notations/Concepts

A forest is a graph where each node has at most one parent. A connected graph is one in which there is a path between every pair of nodes. A tree is a connected forest. A polytree is a "singly" connected graph. That is, there is at most one path from any node to any other node. (Note that trees are a special case of polytrees.) Trees and polytrees will both play an important role in inference.

6.438 Algorithms for Inference

Fall 2014