Categorizing Directed Edges in MAGs

Giorgos Borboudakis AM 646
HY583 Project Spring 2010-2011
Computer Science Department, University of Crete

1 Bayesian Networks

A **Bayesian network** consists of a graphical model and a set of probabilities. The graphical model is a directed acyclic graph (DAG), in which nodes represent random variables and edges represent probabilistic dependence. The terms node and variable will be used interchangeably. Figure 1 shows an example of a Bayesian network. For each node there is a probability distribution on that node given the state of its parents. Thus, a Bayesian network specifies a joint probability distribution over all nodes. We will not go into detail about the joint probability distribution, but will only focus on the structure of Bayesian networks. Before doing this, we will first introduce a few basic definitions. A good introduction to Bayesian networks can be found in [1].

1.1 Basic Definitions

We denote a variable with an upper-case letter (e.g. X) and a set of variables by upper-case bold letters (e.g. $\mathbf{Z}$).

A DAG consists of a set of nodes $\mathcal{V}$, and a set of edges $\mathcal{E}$ between nodes of $\mathcal{V}$. The number of nodes is $n = \mid V \mid$ and the number of edges is is $m = \mid E \mid$, where $\mid \cdot \mid$ denotes set cardinality. A DAG contains only **directed** edges ($\rightarrow$). The two ends of an edge we call **marks** or **orientations**. A directed edge consists of a **tail** ($-$) and an **arrowhead** ($>$). We say an edge is **into** (or **out of**) a node if the edge mark at the node is an arrowhead (or a tail).

Given a DAG $\mathcal{G}$, two nodes are said to be **adjacent** in $\mathcal{G}$ if there is an edge between them. Given two adjacent nodes X and Y, X is said to be a **parent** of Y and Y a **child** of X if $X \rightarrow Y$. A **path** in $\mathcal{G}$ is a sequence of distinct nodes $\langle V_0, \ldots, V_n \rangle$ such that for $0 \leq i \leq n-1$, V_i and V_{i+1} are adjacent. A **directed path** from V_0 to V_n in $\mathcal{G}$ is a sequence of distinct nodes $\langle V_0, \ldots, V_n \rangle$ such that for $0 \leq i \leq n-1$, V_i is a parent of V_{i+1}. X is called an **ancestor** of Y and Y a **descendant** of X if $X = Y$ or there is a directed path from X to Y. A **directed cycle** occurs in $\mathcal{G}$ if $Y \rightarrow X$ and there is a directed path from X to Y. Obviously a DAG cannot contain any directed cycles.

1.2 The Markov Condition and the D-separation Criterion

A Bayesian network represents independence relationships between variables. An edge denotes dependence, whereas the absence of an edge denotes (condi-

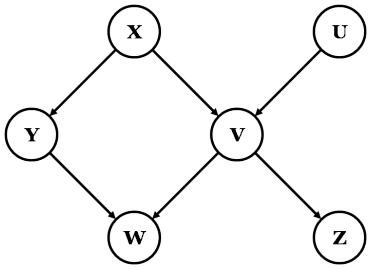

Figure 1: A Bayesian Network

tional) independence. We define conditional dependence (independence) of two sets of variables $\mathbf{X}$,$\mathbf{Y}$ given $\mathbf{Z}$ as $Dep(\mathbf{X};\mathbf{Y}|\mathbf{Z})$ ($Ind(\mathbf{X};\mathbf{Y}|\mathbf{Z})$), where $\mathbf{X}$,$\mathbf{Y}$ are distinct non-empty subsets of the variables $\mathcal{V}$ and $\mathbf{Z}$ is a (possible empty) subset of the variables $\mathcal{V} \setminus \{\mathbf{X}, \mathbf{Y}\}$, where $\setminus$ denotes set difference.

This independence relationships in a Bayesian network are given by the **Markov condition**:

Definition 1. *Any node is conditionally independent of its nondescendants, given its parents.*

For example, consider the Bayesian network shown in Figure 1. The Markov Condition implies:

1. $Ind(Y; \{V, U, Z\} \mid X)$
2. $Ind(W; \{X, U, Z\} \mid \{V, Y\})$
3. $Ind(V; Y \mid \{X, U\})$
4. $Ind(Z; \{X, Y, W, U\} \mid V)$

A graphical criterion called **d-separation** captures exactly all the independencies that are implied by the Markov condition. Before defining d-separation, we first introduce the notion of colliders and blocked paths.

Definition 2. *A node X of a path p is a **collider** if the previous and next nodes of X on the path are into X.*

In Figure 1, the node W is a collider on the path $\langle Y, W, V \rangle$.

Definition 3. *A path p from X to Y is **blocked** by a set of nodes $\mathbf{Z}$, if some node on p:*

1. *is a collider and neither it or any of its descendants are in $\mathbf{Z}$, or*
2. *is not a collider and is in $\mathbf{Z}$*

*If a path p is not blocked by a set of nodes $\mathbf{Z}$, it is said to be **active**.*

In Figure 1, the path $\langle Y, W, V \rangle$ is blocked by $\mathbf{Z} = \emptyset$, because W is a collider and is not in $\mathbf{Z}$.

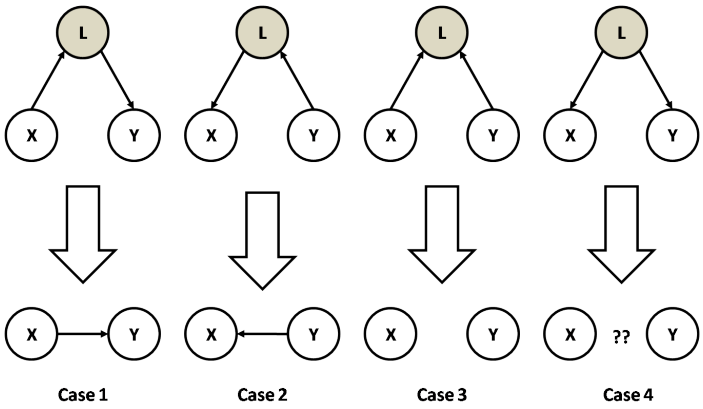

Figure 2: X,Y are the observed variables, whereas L is the unobserved variable.
Case 1: X is a parent of Y (in the context of the measured variables).
Case 2: Y is a parent of X (in the context of the measured variables).
Case 3: X and Y appear to be independent, which is correct.
Case 4: There is no Bayesian Network that can represent this situation; X and Y are not independent, but no edge between them correctly represents their independence relation.

Definition 4. *Two nodes X and Y are* ***d-separated*** *by* $\boldsymbol{Z}$ *if and only if every path from X to Y is blocked by* $\boldsymbol{Z}$*. If two nodes are not d-separated, they are* ***d-connected****.*

Again, in Figure 1, the d-separation criterion implies:

1. $Ind(U; \{X, Y\} \mid \emptyset)$
2. $Ind(U; \{W, Z\} \mid V)$
3. $Ind(Y; \{V, Z\} \mid X)$
4. $Ind(X; \{Y, V\} \mid \{W, Z\})$

1.3 Causal Sufficiency

If we could observe all variables in a domain, we could model the independence relationships between them using a Bayesian network (this is not always the case; we have to make a few (reasonable) assumptions, but this is out of the scope of the current report). However, in practice it is impossible to measure all variables in a domain. Therefore, when learning Bayesian networks, we assume that there are no unobserved variables. This assumption is called the **Causal Sufficiency assumption**.

For two variables and an unobserved (hidden) variable, there are four distinct cases. All distinct cases are shown in Figure 2. Only one of them cannot be modeled by Bayesian networks (the case where both variables have a **hidden common cause**). In the next section we introduce Maximal Ancestral Graphs, which are able to model this.

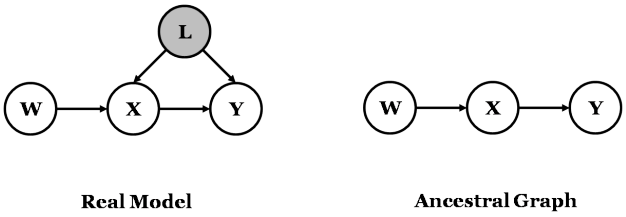

Figure 3: **Left**: Real causal model. **Right**: An ancestral graph representing the model on the left.

2 Maximal Ancestral Graphs

Maximal Ancestral Graphs (MAGs) ([5], [6], [4]) are a class of graphical models with the property of modeling hidden common causes without explicitly introducing them. From now on, the terms hidden common cause and hidden variable will be used interchangeably. Hence, the Causal Sufficiency assumption can safely be dropped. In the following subsection we introduce basic definitions for MAGs.

2.1 Basic Definitions

Most of the definitions for Bayesian networks also hold for MAGs. In addition, we will introduce some definitions for MAGs.

A directed **mixed graph** is a graph which in addition to directed edges ($\rightarrow$) also contains bi-directed edges ($\leftrightarrow$). Notice that a bi-directed edge is not the same as two directed edges between two nodes; bi-directed edges model hidden variables between the two variables. A mixed graph can also contain undirected edges ($-$), but for the purpose of this report we focus only on directed mixed graphs (i.e. mixed graphs without any undirected edges). Again, as in DAGs, between two nodes there is at most one edge. X is called a **spouse** of Y if $X \leftrightarrow Y$ is in $\mathcal{G}$. An **almost directed cycle** occurs when $X \leftrightarrow Y$ is in $\mathcal{G}$ and X is an ancestor of Y. The notions of parent, child, path, directed path, ancestor, descendant and directed cycle remain the same as in DAGs.

Definition 5. *A mixed graph is **ancestral** if the following conditions hold:*

1. *there is no directed cycle, and*
2. *there is no almost directed cycle.*

It follows that an arrowhead edge mark denotes non-ancestry (whereas a tail edge mark denotes ancestry in directed ancestral graphs).

The analogous of the d-separation criterion for mixed graphs is the **m-separation** criterion. For directed mixed graphs, the m-separation criterion is equivalent to the d-separation criterion.

As in DAGs, if two variables are m-separated by any set of variables they are also conditionally independent. However, the opposite does not hold for ancestral graphs. For DAGs, if two variables are not adjacent (i.e. conditionally independent), then there is a set of variables that m-separates the two. This does not always hold for ancestral graphs (see Figure 3). In this example, given

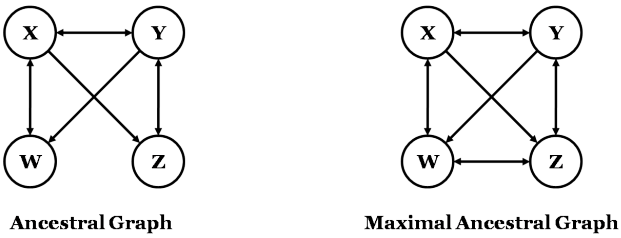

Figure 4: **Left**: An ancestral graph. **Right**: The corresponding maximal ancestral graph.

the model on the left, one could represent it with the ancestral graph on the right. Notice however that W cannot be m-separated from Y by conditioning on any subset of the observed variables. But the ancestral graph does not contain an edge between W and Y, implying that they are independent.

This motivates the following definition:

Definition 6. *An ancestral graph is said to be* ***maximal*** *if every missing edge corresponds to a conditional independency.*

Maximality is closely related to the notion of primitive inducing paths. They are defined as follows:

Definition 7. *Let $\mathcal{G}$ be an ancestral graph. Let X,Y be any two vertices. A path p between X and Y is called a* ***primitive inducing path*** *if:*

1. *every non-endpoint node on p is a collider, and*
2. *every collider on p is an ancestor of either X or Y.*

It is known ([2]) that the presence of a primitive inducing path is *sufficient* and *necessary* for two variables not to be m-separated by any set of other variables in an ancestral graph. Therefore:

Lemma 1. *An ancestral graph is maximal if and only if there is no primitive inducing path between any two non-adjacent nodes in the graph.*

See Figure 4 for an example. The ancestral graph is not maximal, because there is a primitive inducing path between W and Z (W $\leftrightarrow$ X $\leftrightarrow$ Y $\leftrightarrow$ Z, X is an ancestor of Z and Y is an ancestor of W). The ancestral graph can be extended to the maximal ancestral graph on the right by adding a bi-directed edge between W and Z.

2.2 Limitations

If two variables have a directed edge between them and also share a hidden common cause, the MAG will only give us information about the directed edge. That is because edges in MAGs focus on representing ancestral information; a directed edge dominates a bi-directed edge. If we had a bi-directed edge instead of a directed, it would imply that neither of the variables is an ancestor of the other variable, which is obviously false in the presence of a directed edge. But

the situation is even worse: there are cases where even if there is only a hidden common cause between two nodes, the edge has to be oriented towards a node to retain the ancestral semantics of MAGs.

Another issue is that, because MAGs focus on the representation of conditional independencies between variables, there may exist edges in the MAG that do not exist in the real underlying causal model. For this reason, we cannot always interpret directed edges causally.

In the next sections we will analyze those cases and give conditions under which we can (partially) identify them.

3 Pure Causal, Latent and Mixed Edges

A result stemming from the definition of ancestral graphs is the following:

Lemma 2. *An arrowhead end-point out of some node X into another node Y, denotes that Y is not an ancestor of X.*

Proof. Consider a MAG $\mathcal{M}$, and two nodes X and Y.

First, assume $X \rightarrow Y$ is in $\mathcal{M}$. Now assume that Y is an ancestor of X. This would imply that there is a directed path from Y to X. But if there is a directed path from Y to X, $\mathcal{M}$ would contain a directed cycle, which is not possible by definition. Therefore, if $X \rightarrow Y$, Y is not an ancestor of X.

Now assume that $X \leftrightarrow Y$ is in $\mathcal{M}$. Without loss of generality, assume that X is an ancestor of Y. This would imply that there is a directed path from Y to X. But if there is a directed path from Y to X, $\mathcal{M}$ would contain an almost directed cycle, which is not possible by definition. Therefore, if $X \leftrightarrow Y$, Y is not an ancestor of X, and vice versa.

□

A side effect resulting from the semantics of edge end-points is that, if two variables share a directed edge and a hidden common cause, the MAG will only contain a directed edge and will **hide** the bi-directed edge between them. Another side effect is that, because a MAG cannot contain any almost directed cycles, a directed edge in a MAG may not exist in the real underlying causal model, but may be there due to a hidden variable (i.e. it should be bi-directed, but it can't be). Figure 5 shows all possible real models for two nodes that share a directed edge.

A question that arises is, if and under what conditions we can classify directed edges into one (or more) of the three categories. Ideally, we would like to classify each directed edge into one category. However, this is not possible. Directed edges can be classified into the following categories:

1. directed or directed and bi-directed ($\{\rightarrow\}$ or $\{\rightarrow$ and $\leftrightarrow\}$, **mixed**)
2. directed or directed and bi-directed or bi-directed ($\{\rightarrow\}$ or $\{\rightarrow$ and $\leftrightarrow\}$ or $\{\leftrightarrow\}$, **mixed**)
3. directed only ($\{\rightarrow\}$, **pure causal**)

Hence, the problem is defined as follows:

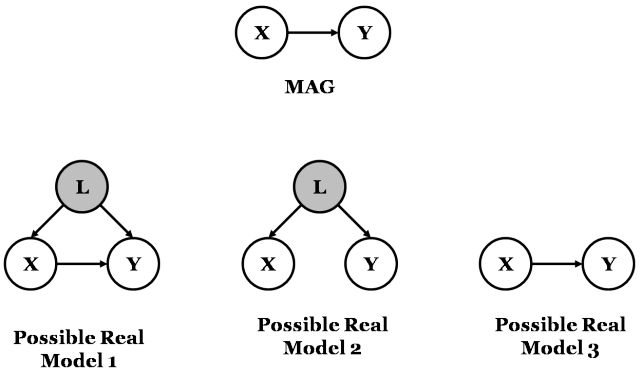

Figure 5: **Left**: X is into Y, and both share a hidden common cause. **Middle**: X and Y share a hidden common cause. **Right**: X is into Y.

Problem 1. *Given a MAG $\mathcal{M}$, classify each directed edge into one of the three categories.*

As a first step towards the solution, recall that a bi-directed edge may be oriented as a directed edge, if and only if it would create an almost directed cycle if oriented as a bi-directed edge. This has to be done to retain the ancestral semantics of edge end-points. This fact leads to a simple rule:

Rule 1. *For a directed edge $X \to Y$, if there is another directed path from X to Y, it may not be directed in the real underlying model, but may be due to a hidden variable. Thus, if there is no directed path from X to Y, the edge surely does not belong to the second category.*

By applying this rule, we can, sometimes, exclude the second case. But we can do even better. The notion of visibility ([3]) will help us at our task. But before defining it, we first define what a collider path is.

Definition 8. *A **collider path** between two nodes X and Y, is a path where each node on it is a collider.*

Now we can define visible and invisible edges:

Definition 9. *A directed edge $X \to Y$ is **visible** if there is a node W not adjacent to Y, such that either there is an edge between W and X that is into X, or there is a collider path between W and X that is into X and every node on the path is a parent of Y. Otherwise the edge is said to be **invisible**.*

Figure 6 shows all possible cases for a visible edge. Visibility is defined in the context of a totally different problem, but seems to be closely related to our problem .

Lemma 3. *If a directed edge is visible, it is also pure causal.*

Proof. We will prove it for one of the four cases and generalize it to all of them. Consider the upper-left case in Figure 6. The edge $X \to Y$ is visible, by definition, subject to W. We will show that the edge is also pure causal, i.e. there is no hidden common cause of X and Y.

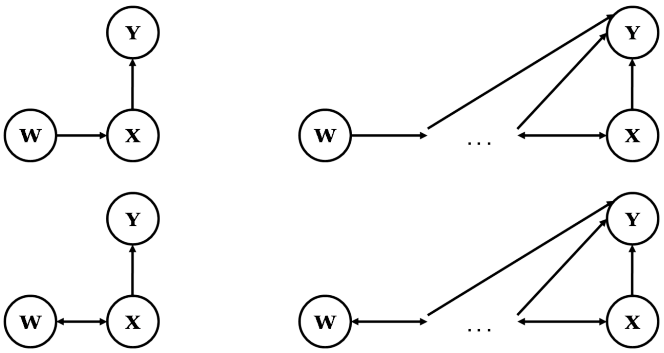

Figure 6: All possible cases where the edge $X \to Y$ is visible.

Assume the real model contains the directed edge $X \to Y$ and X, Y share a hidden common cause L. By observing the MAG, we see that there is no edge between W and Y, which means that they are m-separated by $\{\emptyset\}$ or $\{X\}$. But $\{\emptyset\}$ cannot m-separate W and Y, because the path $W \to X \to Y$ remains active. Neither can $\{X\}$, because it opens the path $W \to X \leftarrow L \to Y$. But if it cannot be m-separated by any set of measured variables, W and Y must be connected. Contradiction.

But we also saw that if $X \to Y$ and there is another directed path from X to Y, the edge may not be directed at all in the real model. Therefore, we also have to show that even in that case, $X \to Y$ is pure causal, if it is visible.

Assume that X, Y share a hidden common cause L in the real model and there is an indirect directed path p from X to Y (i.e. they are not directly connected via a directed edge). Again, because there is no edge between W and X, they have to be m-separated by $\{\emptyset\}$ or some subset of the nodes on $p \bigcup X$. But $\{\emptyset\}$ cannot m-separate W and Y, because the path $W \to X \to Y$ remains active. Neither can any subset of $p \bigcup X$, because it opens the path $W \to X \leftarrow L \to Y$. That is because if X (which is a collider on that path) or any of its descendants is in the separating set, it opens the path $W \to X \leftarrow L \to Y$. But all nodes on p are descendants of X. Thus, W and Y cannot be m-separated by set of measured variables and must be connected. Contradiction.

Now to generalize this proof, notice that what holds for the upper-left case, also holds for the lower-left case. Also notice that for the other cases, to block all paths from W to Y, one has to include all nodes on the collider path in the separating set. That is because the first node V_1 has to be there, to block the path $W \to V_1 \to Y$, but it opens the path $W \to V_1 \leftrightarrow V_2 \to Y$, thus V_2 has to be also in the separating set, which inductively holds for all nodes on p until X. Because of that, the upper-right and lower-right cases trivially reduce to the upper-left and lower-left cases, by including all nodes on p in the conditioning set. □

Based on lemma 3, we can define the following rule:

Rule 2. *For a directed edge $X \to Y$, if the edge is visible, the edge belongs to the third category.*

Algorithm 1: Find Visible Edges(MAG $\mathcal{M}$)

```
Mark all directed edges as invisible;
Q ← Queue();
for each directed edge X → Y in M do
    if ∃ W s.t. (W → X or W ↔ X) and (W not adjacent to Y) then
        Mark X → Y as visible;
        Q.enqueue({X, Y});
while not Q.isempty() do
    {X, Y} ← Q.dequeue();
    for each V s.t. X ↔ V and V → Y do
        if V → Y is invisible then
            Mark V → Y as visible;
            Q.enqueue({V, Y});
return All Marked Edges;
```

To summarize:

Rule 3. *For a directed edge $X \to Y$:*
If the edge is visible
$X \to Y \Rightarrow$ *category 3 ($\{\to\}$)*
Else if there is another directed path from X to Y
$X \to Y \Rightarrow$ *category 2 ($\{\to\}$ or $\{\to$ and $\leftrightarrow\}$ or $\{\leftrightarrow\}$)*
Else
$X \to Y \Rightarrow$ *category 1 ($\{\to\}$ or $\{\to$ and $\leftrightarrow\}$)*

Algorithm *Find Visible Edges* finds all visible edges in a MAG. We will prove that it is sound and complete and provide an upper bound for its time complexity. The following lemma will be helpful.

Lemma 4. *Given a collider path p $\langle W* \to V_1 \leftrightarrow V_2 \leftrightarrow \cdots \leftrightarrow V_k \rangle$, (where $* \to$ can be either $\to$ or $\leftrightarrow$) and a node Y s.t. W and Y are not adjacent and for each node $V_i, i = 1, 2, \ldots, k$, V_i is into Y, all edges $V_i \to Y$ are visible.*

Proof. By definition, $V_1 \to Y$ is visible, because W is not adjacent to Y. We will prove that this also holds for any other edge $V_i \to Y$ by induction.
Base case: $V_2 \to Y$ is visible relative to W, because $V_1 \leftrightarrow V_2$ is a collider path and each node on the path is into Y
Inductive hypothesis: $V_i \to Y$ is visible relative to W
Inductive step: $V_{i+1} \to Y$ is visible relative to W, because:

1. W is not connected to Y,
2. $W* \to V_1$,
3. there is a collider path from V_1 to V_i, and every node on it is into Y,
4. by adding V_{i+1} to the path, it remains a collider path because $V_i \leftrightarrow V_{i+1}$ and each node is into Y because V_{i+1} is also into Y.

□

Lemma 4 introduces the basic idea which is used in algorithm *Find Visible Edges*. Starting from a trivial visible edge (without the collider path), it incrementally searches for additional visible edges. The most important idea to make the algorithm efficient is that, if while searching for visible edges we find an edge that already is marked as visible, the current search can be stopped. That is because if we encounter such an edge, it means that we already searched out of all collider paths from that node and all visible edges after that will already be marked as visible. Because of that, the search tree can be pruned so much making the algorithm run in $O(n \cdot m)$ time. We next show that the algorithm *Find Visible Edges* is sound and complete and that its time complexity is $O(n \cdot m)$.

Theorem 1. *The algorithm Find Visible Edges is* ***sound*** *and* ***complete****, i.e. it finds all visible edges imposed by definition 9.*

Proof. Obviously the algorithm is sound, because an edge will be classified as visible if and only if it is due to some of the cases in definition 9, which are the cases considered by the algorithm.

Also the algorithm is complete, because the algorithm first finds all trivial visible edges (first two cases) and then incrementally searches for all visible edges resulting from collider paths (which is correct due to lemma 4). This search finds all possible visible edges for the last two cases. Thus, the algorithm always finds all possible visible edges as defined in definition 9. □

Theorem 2. *The algorithm Find Visible Edges runs in $O(n \cdot m)$ time.*

Proof. The computational costly parts of the algorithm are lines 3-6 and 7-12. Note that each operation on the queue takes constant time $O(1)$.

The loop at line 3 will run $O(m)$ times. For the if statement at line 4, we have to find all nodes $W \rightarrow X$ and $W \leftrightarrow X$, which can be computed in $O(n)$ time (each node is adjacent to at most $n-1$ other nodes), and then out of those check if any is not adjacent to Y, which again takes $O(n)$ time. The statements at lines 5 and 6 take constant time. Thus, lines 3-6 take $O(n \cdot m)$ time.

At this point, Q may contain at most $O(m)$ elements. At each iteration we remove one element (line 8). Line 9 takes $O(n)$ time (similar to line 4). Lines 10-12 take constant time. The only line that could cause a problem is line 12, because it increases the elements in Q and could lead to more iterations. Notice however that an element is added only if a visible edge is found which was previously invisible (line 10). But if it was invisible, it never was in Q. Therefore each visible edge will be **exactly once** in Q. But the number of visible edges is at most $O(m)$. Thus, the outer loop (line 7) will run $O(m)$ times, with a cost of $O(n)$ each, and the total running time of lines 7-12 is $O(n \cdot m)$.

Both, lines 3-6 and 7-12 run in $O(n \cdot m)$ time, therefore the algorithm *Find Visible Edges* runs in $O(n \cdot m)$ time. □

Theorem 3. *The total time needed to classify all edges using Rule 3 is $O(n^2 + n \cdot m)$.*

Proof. The time to find all ancestors for some node is $O(n+m)$, by using some modified graph search procedure such as DFS or BFS. This has to be done for each node and therefore takes $O(n^2 + n \cdot m)$ time. Classifying all edges takes $O(n \cdot m)$ time. Therefore the total time complexity is $O(n^2 + n \cdot m)$. □

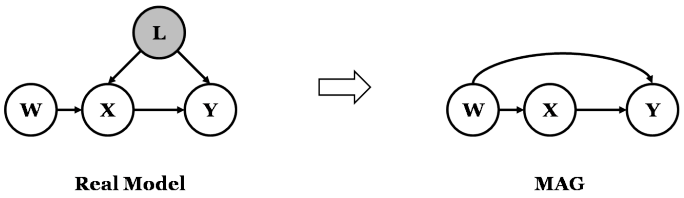

Figure 7: **Left**: Real underlying model. **Right**: Respective MAG.
The MAG contains the directed edge $W \to Y$ even though it does not exist in the real model. That happens because it W and Y cannot be m-separated by conditioning on any subset of the observed variables.

4 Direct and Possibly Indirect Edges

As stated in section 2.2, there may exist edges in a MAG, without existing in the real underlying causal model. That happens because the MAG focuses on representing independence relationships between variables rather than causal relationships. The consequence of this is that directed edges cannot always be interpreted causally. An example is shown in Figure 7.

It is proven ([2]) that two nodes are not m-separated by any set of variables if and only if there is an inducing path between them. Therefore, an extra edge between two nodes exists in a MAG if and only if there is an inducing path between them. In Figure 7, the inducing path is $W \to X \leftarrow L \to Y$. But how are the edges directed? This is answered by the next theorem ([5]):

Theorem 4. *If $\mathcal{G}$ is an ancestral graph then there exists a unique maximal ancestral graph $\mathcal{M}$ formed by adding bi-directed edges to $\mathcal{G}$.*

So, additional edges in a MAG are bi-directed. But that is not always the case; recall that sometimes a bi-directed edge has to be directed because it would create an almost directed cycle if left as bi-directed. Based on this, we can define the next rule.

Rule 4. *A directed edge $X \to Y$ which does not exist in the real model exists in the MAG if and only if, a) there is an inducing path between X to Y, and b) there is a directed path from X to Y. Thus, a directed edge **possibly** does not exist in a MAG if and only if both a) and b) hold, and should be classified as **possibly indirect**. If this is not the case, the edge exists and should be classified as **direct**.*

An immediate problem that occurs is that it is very hard to identify inducing paths when only observing the MAG. That is, because bi-directed edges may be hidden by directed edges. But we already solved this problem in section 3. Although we may not be able to fully identify in which category each directed edge belongs, we can at least partially identify it. A directed edge is **possible bi-directed** if it is in category 2 or 3. A path is a **possible inducing path**, if it would be an inducing path if possible bi-directed edges would be treated as bi-directed. The next rule describes when a directed edge is direct or possibly indirect.

Rule 5. *For a directed edge $X \to Y$:*
If $\exists$ possible inducing path from X to $Y \wedge \exists$ directed path from X to Y
$X \to Y \Rightarrow$ *possibly indirect*
Else
$X \to Y \Rightarrow$ *direct*

Theorem 5. *The complexity of applying rule 5 for all directed edges in a MAG $\mathcal{M}$, is $O(n^2 + n * m)$.*

Proof. As a first step, we have to find the category of each edge. This takes $O(n \cdot m)$ time. Then we have to find all descendants for each node, and all nodes it may have an inducing path to. Finding the descendants of one node can be done using a modified graph search procedure such as DFS or BFS in $O(n+m)$ time. The same applies for finding all nodes it may have an inducing path to, by modifying the search procedure accordingly. There are a total of n nodes, therefore we need $O(n^2 + n \cdot m)$ time.

The second step is to classify each edge. For a single edge $X \to Y$ it takes $O(n)$ time, because we just have to check if Y is in both, the descendant and possible inducing path sets, which can be done with a linear search. This is done for each directed edge (at most m), therefore the time complexity is $O(n \cdot m)$.

The total time complexity of applying rule 5 is $O(n^2 + n \cdot m)$. □

References

[1] Richard E. Neapolitan; Learning Bayesian Networks; Prentice Hall, 2003.

[2] Jiji Zhang; On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias; Elsevier - Artificial Intelligence, 2008.

[3] Jiji Zhang; Causal Reasoning with Ancestral Graphs; Journal of Machine Learning, 2008.

[4] Jiji Zhang; Causal Inference and Reasoning in Causally Insufficient Systems; PhD Thesis, Department of Philosophy, Carnegie Mellon University, 2006.

[5] Thomas Richardson and Peter Spirtes; Ancestral Graph Markov Models; The Annals of Statistics, 2002.

[6] Peter Spirtes, Christopher Meek and Thomas Richardson; Causal Inference in the Presence of Latent Variables and Selection Bias; Eleventh Conference on Uncertainty in Artificial Intelligence, 1995.