Graphical Representation of Independence Structures

Kayvan Sadeghi

Jesus College

University of Oxford

Department of Statistics

A thesis submitted for the degree of Doctor of Philosophy

Trinity Term 2012

Abstract

In this thesis we describe subclasses of a class of graphs with three types of edges, called loopless mixed graphs (LMGs). The class of LMGs contains almost all known classes of graphs used in the literature of graphical Markov models. We focus in particular on the subclass of ribbonless graphs (RGs), which as special cases include undirected graphs, bidirected graphs, and directed acyclic graphs, as well as ancestral graphs and summary graphs. We define a unifying interpretation of independence structure for LMGs and pairwise and global Markov properties for RGs, discuss their maximality, and, in particular, prove the equivalence of pairwise and global Markov properties for graphoids defined over the nodes of RGs. Three subclasses of LMGs (MC, summary, and ancestral graphs) capture the modified independence model after marginalisation over unobserved variables and conditioning on selection variables of variables satisfying independence restrictions represented by a directed acyclic graph (DAG). We derive algorithms to generate these graphs from a given DAG or from a graph of a specific subclass, and we study the relationships between these classes of graphs. Finally, a manual and codes are provided that explain methods and functions in R for implementing and generating various graphs studied in this thesis.

Keywords: Ancestral graph, bidirected graph, chain graph, composition property, directed acyclic graph, global and pairwise Markov property, graphoid, independence model, marginalisation and conditioning, maximality, MC graph, separation criterion, summary graph, undirected graph.

DECLARATION BY CANDIDATE

I hereby declare that this thesis is my own work and effort and that it has not been submitted anywhere for any award. Where other sources of information have been used, they have been acknowledged.

Signature: .....

Date: .....

ACKNOWLEDGMENTS

First and foremost, I would like thank my supervisor Professor Steffen Lauritzen for providing me with the advice and guidance without which the completion of this thesis would not have been possible. I am deeply grateful to Professor Nanny Wermuth who made it possible for me to travel to Sweden on various occasions while writing this thesis and who spent countless hours with me discussing the material. Her time, suggestions, and kindness have proved invaluable. I would also like to thank Professor Sir David Cox for the comments he made on this thesis at all its different stages and for the general discussions we were able to have together. Professor Giovanni Marchetti provided me with helpful ideas on R codes and has been available throughout the writing of this thesis to assist along the way. Professor Thomas Richardson also provided me with valuable comments and was very helpful with my questions, for which I am most grateful. I must also thank Dr Ruth Ripley who gave me a helping hand with all my R problems. I should also like to thank Jesus College for providing me with the welcoming environment in which this thesis was completed.

I owe much to my family whose encouragement and support throughout my studies have enabled me to reach this stage. Mahdad and Azita in particular showed me great kindness and provided me with a home away from home in Sweden and I am greatly thankful to them. My time at Oxford was made enjoyable thanks to the many friends I made here as well as the support of all my friends back in Iran and elsewhere. Finally, I would like to thank Tanya who proofread my thesis and whose presence in my life made the process a much happier one.

Contents

1 Introduction and background **1**

1.1 Introduction 1

1.2 Summary of contributions 4

1.3 Basic definitions and concepts 7

1.3.1 Graph theoretical definitions and abstract independence models 7

1.3.2 Probabilistic conditional independence models 15

2 Independence structures for mixed graphs **19**

2.1 Introduction to loopless mixed graphs 19

2.2 Ribbonless graphs and their relationship to MC graphs 28

2.3 Subclasses of loopless mixed graphs 38

2.4 Maximality for ribbonless graphs 45

2.5 Markov properties for ribbonless graphs 57

2.6 Summary and future work 67

3 Stable mixed graphs **68**

3.1 Marginalisation, conditioning, and stability 68

3.2 Ribbonless graphs 73

3.2.1 Generating ribbonless graphs 74

3.2.2 Two necessary properties of α_{RG} 86

3.3 Summary graphs 91

3.3.1 Generating summary graphs 91

3.3.2 Two necessary properties of α_{SG} 95
3.4 Ancestral graphs 99
3.4.1 Generating ancestral graphs 99
3.4.2 Two necessary properties of α_{AG} 111
3.5 The relationship between different types of stable mixed graphs . 112
3.5.1 Corresponding types of stable mixed graphs 114
3.5.2 Properties of functions generating stable mixed graphs . . 115
3.5.3 Discussion on different types of stable mixed graphs 118
3.6 Summary and future work 122

4 Loopless mixed graphs in R **123**
4.1 Introduction 123
4.2 Defining directed acyclic graphs 124
4.3 Defining mixed graphs 128
4.4 Generating stable mixed graphs 130
4.5 Maximality and m-separation 133

Glossary **136**

Bibliography **143**

Chapter 1

Introduction and background

1.1 Introduction

Introduction and motivation. Graphical Markov models have become widely used in various forms and various subjects in recent years. These models use graphs to represent conditional independence statements of sets of random variables. Nodes of the graph correspond to random variables and edges typically capture dependencies.

For this purpose, several classes of graphs with various interpretations of independencies have been described in the literature. These range from the class of simple undirected graphs with simple graph theoretical separation for interpretation of independencies [Lauritzen, 1996] to the various forms of mixed graphs [Richardson and Spirtes, 2002], including chain graphs with several different interpretations of independencies [Drton, 2009].

In spite of the significant differences among these graphs, their structural similarities are considered sufficient to motivate an attempt to unify these in a class of graphs which will be called loopless mixed graphs (LMGs). By studying the independence structure and properties of LMGs, many of the known and unknown properties of its subclasses can be implied.

In particular, the focus of our study is on a largest subclass of loopless mixed

graphs, which we shall term *ribbonless*. We define a unifying interpretation of the conditional independence of a pair of arbitrary sets of nodes given another set, corresponding to the so-called global Markov property, as well as a unifying interpretation of a missing edge in the graph, corresponding to the so-called pairwise Markov property. We also give conditions for these interpretations to be equivalent.

One of the most important subclasses is the class of directed acyclic graphs (DAGs) [Kiiveri et al., 1984]. Their corresponding Markov models, often known under the name of Bayesian networks [Pearl, 1988], have direct applications to a wide range of areas including econometrics, social sciences, and artificial intelligence. For instance however, when unmeasured latent variables occur, one can in general no longer capture the implied independence structure among observed variables by a DAG. In this sense the DAG structures are not stable under marginalisation. A similar problem occurs because DAG structures are not stable under conditioning [Cox and Wermuth, 1996].

This makes it necessary to identify and study a class of graphs that includes DAGs and is stable under marginalisation and conditioning in the sense that it is able to express the induced independence model after marginalisation and conditioning through an object of the same class. The methods that have been used to solve this problem employ three different types of edges instead of a single type. Three known classes of graphs have previously been suggested for this purpose in the literature. We call these stable mixed graphs (under marginalisation and conditioning) and they include MC graphs, summary graphs (SGs), and ancestral graphs (AGs). Studying the similarities, differences, and relationships among these three classes enables us to decide which class should be used for a particular purpose.

Structure of the thesis. In the next section we give a literature review of contributions by other authors and a summary of the results presented in this thesis. In Section 1.3.1 we define the basic concepts of graph theory, independence models, which contain probabilistic independence models as a special case, separation criteria on graphs, and compositional graphoids. We then, in Section 1.3.2, relate them to the concept of probabilistic conditional independence.

In Chapter 2, we formally define the class of LMGs, and associate a separation criterion, called m-separation, to this class in Section 1. Using m-separation, the class of LMGs is shown to contain almost all known classes of graphs in the literature as its subclasses (Section 3). In Section 2, we define the class of ribbonless graphs (RGs), which is a modification of the known class of MC graphs to use m-separation, discuss some motivations behind defining this subclass, and precisely relate them to MC graphs. In Section 3, we introduce almost all subclasses of LMGs discussed in the literature of graphical models. In Section 4, we introduce the concept of maximality by demanding that an additional edge will change the independence model. We discuss maximality of RGs and its subclasses, and show how to generate maximal graphs that induce the same independence model from non-maximal graphs. In Section 5, we define pairwise and global Markov properties for the independence models defined over RGs, and prove that, for maximal graphs and compositional graphoid independence models, pairwise and global Markov properties are equivalent.

In Chapter 3, Section 1 we formally define marginalisation and conditioning for independence models in such a way that it conforms with marginalisation and conditioning for probability distributions. We also formally define a class of graphs to be stable after marginalisation and conditioning. In Section 2, we define and study algorithms to generate new RGs from given DAGs or RGs (with their induced independence models) and two subsets of the node set that are going to

be marginalised over and conditioned on. We prove that the given algorithms generate a stable class of graphs. In Sections 3 an 4, we treat summary and ancestral graphs as subclasses of RGs and define a parallel theory for them. We define algorithms to generate them from DAGs or from graphs of the same class and prove that such algorithms generate stable mixed classes. In Section 5, we study the relationships between the three types of stable mixed graphs. For this purpose, we consider the RG-, SG-, and AG-generating algorithms to be functions from the class of DAGs, and we prove that the functions are surjective. In addition, we give a discussion on the use of different types of stable mixed graphs.

In Chapter 4, we give a manual for using the implementation of introduced algorithms and the conditions of some of the theorems in R.

1.2 Summary of contributions

Summary of previous contributions. Graphical models can be traced back to Gibbs [1902], who used models corresponding to undirected graphs in statistical physics, and Wright [1934], who used directed graphs for establishing path analysis in genetics.

The ideas of using undirected graphs were developed mostly under the name of random fields, e.g., see Spitzer [1971]. Such ideas together with the by then growing theory on contingency tables, e.g., Goodman [1970] led to the formal exposition of graphical models using undirected graphs in Darroch et al. [1980]. Meanwhile, the study of statistical models that treat all variables on equal standing for Gaussian random variables (and for categorical variables independently) revealed the important property that if the ij array of concentration matrix is zero then i is independent of j given the rest of variables [Dempster, 1972]. This led to the use of undirected graphs for representing the independence structure among normally distributed variables, e.g., in Hodapp and Wermuth [1983].

On the other hand, the ideas of path analysis received attention later in economics, e.g., in Wold [1954], and in the social sciences, e.g., in Blalock [1971]. Further developments of such ideas led to the study of DAGs (as a subgraph of a larger mixed graph called recursive causal graph) in Kiiveri et al. [1984] for causal interpretations. For DAGs, Pearl [1988] introduced a method for identifying conditional independence for subsets of the node set, called the d-separation criterion.

The concepts of pairwise and global Markov properties for undirected graphs were introduced in the unpublished paper Hammersley and Clifford [1971] in the context of random fields and their equivalence for positive densities, -later known as the Hammersley–Clifford theorem-, was proven. Simpler proofs were given later independently by several authors, e.g., Grimmett [1973], Besag [1974]; see also Clifford [1990]. In the developments of abstract theory of conditional independence one variant of the Hammersley–Clifford theorem was proven in Pearl [1988] for graphoids, which are abstract independence models with four standard properties of independencies of any multivariate distribution together with an additional property satisfied by distributions with positive densities; see also Studeny [1989] and Geiger et al. [1990]. Conditional independence models for undirected graphs were discussed comprehensively in Chapter 3 of Lauritzen [1996].

For the purpose of obtaining different interpretations of conditional independence structure, other types of graphs have been also used in the literature. The idea of using graphs containing mixed types of edges was first introduced in Lauritzen and Wermuth [1989] who defined chain graphs for the conditional independence interpretation of mixed variables. The bidirected graph models and their independence model were discussed by Kauermann [1996]. Bidirected chain graphs were introduced and studied by Cox and Wermuth [1993] and in Wermuth and Cox [2004], under the name of multivariate regression chains. For discrete vari-

ables, more attractive features of regression graph models were derived recently; see Drton [2009], who speaks of chain graph models of type IV.

For DAG models with latent and selection variables, the question of finding graphical models to capture the modified independence model after marginalisation and conditioning was first posed in Wermuth et al. [1994] and Cox and Wermuth [1996]. To answer this problem three types of graphs have been introduced, even though, in addition to this main goal, each of them has been equipped to deal with different needs. These are MC graphs, defined and studied by Koster [2002], summary graphs, introduced by Wermuth et al. [1994] and recently developed further by Wermuth [2011], and (maximal) ancestral graphs, defined and studied by Richardson and Spirtes [2002].

The popularity of graphical models in recent years has been largely due to the evolution of modern computers, which provide fast ways of handling the data used in such models. Several packages for graphical models have been provided in the statistical computing programming language R. A list of available packages with some descriptions can be found at CRAN Task View gRaphical Models in R on http://cran.r-project.org/web/views/gR.html.

Contributions of the present thesis. In this thesis, two series of results have been obtained. These are as follows:

1) In Chapter 2:

- unifying the Markov theory of a number of different types of graphs used in graphical models into LMGs with the unifying m-separation criterion;
- examining maximality of RGs, which is a subclass of these graphs, and its subclasses, and generating maximal graphs of the same type that induce the same independence model;
- defining a pairwise Markov property for maximal RGs, and proving that if

compositional graphoid axioms hold then the defined global and pairwise Markov properties are equivalent.

2) In Chapter 3:

- providing clear and fast algorithms for generating stable mixed graphs from DAGs or graphs of the same type;
- establishing a parallel theory for the three types of stable mixed graphs;
- studying the relationship between the graphs using the characteristics and properties of the generating algorithms.

The algorithms and criteria have been implemented in R and are now available in package `ggm`.

A major part of Chapter 2 has been submitted for publication in Sadeghi and Lauritzen [2011]. A major part of Chapter 3 has been accepted for publication in Sadeghi [2012]. A manual for the implemented R functions has been accepted for publication subject to some corrections in Sadeghi and Marchetti [2011].

1.3 Basic definitions and concepts

1.3.1 Graph theoretical definitions and abstract independence models

In this section we introduce definitions and notations to describe independence models, graphs, different types of paths, separation criteria, and compositional graphoid axioms. Some of the graph theoretical definitions have been quoted from West [2001]. For further discussion on independence models, see Studeny [2005].

Independence models and graphs. An *independence model* $\mathcal{J}$ over a set V is a set of triples $\langle X, Y \mid Z \rangle$ (called *independence statements*), where X, Y, and

Z are disjoint subsets of V and Z can be empty, and $\langle \varnothing, Y \mid Z \rangle$ and $\langle X, \varnothing \mid Z \rangle$ are always included in $\mathcal{J}$. The independence statement $\langle X, Y \mid Z \rangle$ is interpreted as "X is independent of Y given Z".

A *graph* G is a triple consisting of a *node* set or *vertex* set V, an *edge* set E, and a relation that with each edge associates two nodes (not necessarily distinct), called its *endpoints*. When nodes i and j are the endpoints of an edge, these are *adjacent* and we write $i \sim j$. We say the edge is *between* its two endpoints. We usually refer to a graph as an ordered pair $G = (V, E)$. Graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are called *equal* if $(V_1, E_1) = (V_2, E_2)$. In this case we write $G_1 = G_2$.

Notice that the graphs that we use in this thesis and, in general, in the context of graphical models are so-called *labeled graphs*, i.e. every node is considered a different object. Hence, for example, graph i —— j —— k is not equal to j —— i —— k.

We use the notation $\mathcal{J}^G$ for an independence model defined over the node set of G. Among the independence models over the node set V of a graph G, those that are of interest to us *conform* with G, meaning that $i \sim j$ in G implies $\langle i, j \mid C \rangle \notin \mathcal{J}^G$ for any $C \subseteq V \setminus \{i, j\}$. Henceforth, we assume that independence models $\mathcal{J}^G$ conform with G, unless otherwise stated.

For example, in graph G in Figure 1.1, $\mathcal{J}^G = \{\langle i, l \mid j \rangle, \langle i, k \mid \varnothing \rangle\}$ conforms with G whereas $\mathcal{J}^G = \{\langle i, l \mid j \rangle, \langle i, j \mid \varnothing \rangle\}$ does not conform with G because of $\langle i, j \mid \varnothing \rangle$ independence statement. Notice that we use the notation i instead of $\{i\}$ for a subset consisting of a single element i in an independence statement.

Basic graph theoretical definitions. Here we introduce some basic graph theoretical definitions. A *loop* is an edge with the same endpoints. *Multiple edges* are edges with the same pair of endpoints. A *simple graph* has no loops or multiple edges. The number of edges adjacent to a node i is called the *degree* of i.

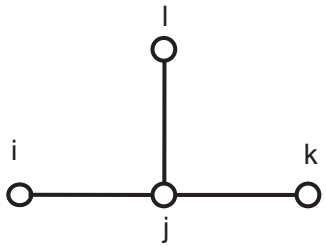


Figure 1.1: A graph G with which $\mathcal{J}^G = \{\langle i, l \,|\, j\rangle, \langle i, k \,|\, \varnothing\rangle\}$ conforms, whereas $\mathcal{J}^G = \{\langle i, l \,|\, j\rangle, \langle i, j \,|\, \varnothing\rangle\}$ does not conform.

If the graph assigns an ordered pair of nodes to each edge then the graph is a *directed graph*. We say that the edge is *from* the first node of the ordered pair *to* the second one. We use an arrow, $j \longrightarrow i$, to draw an edge in a directed graph. We also call j a *parent* of i and i a *child* of j, and we use the notation $\mathrm{pa}(i)$ for the set of all parents of i in the graph.

A *subgraph* of a graph G_1 is graph G_2 such that $V(G_2) \subseteq V(G_1)$ and $E(G_2) \subseteq E(G_1)$ and the assignment of endpoints to edges in G_2 is the same as in G_1. An *induced subgraph* by nodes $N \subseteq V$ is a subgraph that contains all and only nodes in A and all edges between two nodes in N. A subgraph induced by edges $F \subseteq E$ is a subgraph that contains all and only edges in F and all nodes that are endpoints of edges in F.

A *walk* is a list $\langle i_0, e_1, i_1, \ldots, e_k, i_k\rangle$ of nodes and edges such that for $1 \leq n \leq k$, the edge e_n has endpoints v_{n-1} and v_n. A *path* is a walk with no repeated node. If the graph is simple then the path can be determined uniquely by an ordered sequence of node sets. Throughout this thesis, however, we use node sequences to describe paths even in graphs with multiple edges, but we suppose that the edges of the path are all determined. Usually it is apparent from the context or the type of a path which edge belongs to the path in multiple edges.

If $\pi_1 = \langle i = i_0, i_1, \ldots, i_n, h\rangle$ and $\pi_2 = \langle h, j_m, j_{m-1}, \ldots, j_0 = j\rangle$ are paths, their *combination* $\pi_{12} = \pi_1 \circ \pi_2$ is the path $\pi_{12} = \langle i, \ldots, k, j_{q-1}, \ldots, j\rangle$, where

$k = i_p = j_q$ is the first node of π_1 on both paths. If $k = h$ then π_{12} is simply the concatenation of the two paths. In general, the concatenation of two paths will be a walk and not a path as the paths may intersect in more than one point.

We say a path is *between* the first and the last nodes of the list in G. We call the first and the last nodes *endpoints* of the path and all other nodes *inner nodes*.

A graph is *connected* if there exists a path between any pairs of nodes. In general, nodes connected by paths define a *connected component* in the graph. One can observe that a connected graph has only one connected component. In this thesis we mostly focus on connected graphs since all the results presented in this thesis can be easily generalised from a connected component to a graph with several connected components. Thus, by "graph", we always mean "connected graph", unless otherwise stated.

A *subpath* of a path π is a path that can be considered a subgraph of π with the ordering associated with π. A *cycle* in a graph is a simple subgraph whose nodes can be placed around a circle so that two nodes are adjacent if these appear consecutively along the circle.

Directed graphs can have three types of *V-configurations*, i.e. a path with three nodes and two edges: $i \prec\!\!- t \prec\!\!- j$, $i \prec\!\!- s \!\!-\!\!\succ j$, and $i \!\!-\!\!\succ c \prec\!\!- j$, where the inner node is respectively called a *transition* (t), a *source* (s) or a *collision* node (c) on the V-configuration or more generally on a path of which the V-configuration is a subpath. A V-configuration with inner transition, source, or collision node is called, a *transition, source*, or *collision V-configuration* respectively. Transitions or source V-configurations can be also called *non-collision* V-configurations. We may speak of transition, source, or collision nodes without mentioning the V-configuration or path when this is apparent from the context. A V-configuration is called *unshielded* if its endpoints are not adjacent. Notice that originally and in most texts, e.g. in Kiiveri et al. [1984], a V-configuration is by definition un-

shielded. Here we consider a wider range of paths with three nodes and two edges (both shielded and unshielded ones) to be V-configuration.

A path (or a cycle) in a directed graph is *direction-preserving* if all its inner nodes (or nodes) are transition nodes. A directed graph is *acyclic* if it has no direction-preserving cycle.

If in a direction-preserving path an arrow starts at a node j and an arrow points to a node i then j is an *ancestor* of i, and i a *descendant* of j. We use the notation $\mathrm{an}(i)$ for the set of all ancestors of i. Node j is an *A-line ancestor* of node i if all inner nodes in a direction-preserving ij-path are in set A. A path in a directed graph is an *alternating path* if the direction of the arrows changes at each inner node. This implies that no inner node is a transition node and that the inner nodes alternate between source and collision nodes.

A DAG is called the *partial-ancestor graph* with respect to nodes A, denoted by $G^{V.A}_{\mathrm{anc}}$, if it results from the directed acyclic graph G by turning every A-line ancestor into a parent, which is by adding an arrow $i \prec\!\!\!\!\!\!\leftarrow j$ whenever j is an A-line ancestor of i in G. In the partial-ancestor graph $G^{V.A}_{\mathrm{anc}}$, there is an *active alternating ij-path* with respect to A if there is an edge between i and j or there is an alternating ij-path with some inner nodes such that all its source nodes are in A and all its collision nodes are not in A. We also say that the alternating ij-path is *active with respect to A and B* if all its source nodes are in A and all its collision nodes are in B. We say a path is *from i to j* if it is between i and j, and on the path i is a parent and j is a child.

Separation criteria. A *separation criterion* $\mathcal{S}$ is a function that maps $G = (V, E)$ to an independence model $\mathcal{J}_{\mathcal{S}}(G)$ and has the two following properties:

1) It is *path-based* in the sense that for disjoint subsets, A, B, and C of V, $\langle A, B \,|\, C\rangle \in \mathcal{J}_{\mathcal{S}}(G)$ if and only if in G or in a derived graph $H(G; A, B, C) =$

(N, F), where $A \cup B \cup C \subseteq N$, there is no walk of a specific type determined by C between A and B. The type of walk is associated with $\mathcal{S}$ and has the property that adding an edge to graph does not destroy the walk;

2) If there are nodes $i \in A$ and $j \in B$ such that $i \sim j$ then $\langle A, B \,|\, C\rangle \notin \mathcal{J}_{\mathcal{S}}(G)$, for all subsets C disjoint from A and B.

If $\langle A, B \,|\, C\rangle \in \mathcal{J}_{\mathcal{S}}(G)$ then we use the notation $A \perp_{\mathcal{S}} B \,|\, C$ and say that "C *$\mathcal{S}$-separates* A and B". Notice that $\mathcal{S}$ *induces an independence model* $\mathcal{J}_{\mathcal{S}}(G)$ on G by $A \perp_{\mathcal{S}} B \,|\, C \iff \langle A, B \,|\, C\rangle \in \mathcal{J}_{\mathcal{S}}(G)$.

Notice also that condition 2 of the definition implies that the induced independence model $\mathcal{J}_{\mathcal{S}}(G)$ conforms with G. Notice, however, that if $\langle A, B \,|\, C\rangle \notin \mathcal{J}_{\mathcal{S}}(G)$ for all C we cannot conclude that $i \sim j$, for every pair of nodes $i \in A$ and $j \in B$. This is true if G is a so-called maximal graph with respect to $\mathcal{S}$; see Section 2.4.

An example of a separation criterion on DAGs. As an example of a separation criterion, here we recall a separation criterion for DAGs that was given in Marchetti and Wermuth [2009]. Consider a directed acyclic graph $G = (V, E)$ and suppose V is partitioned into four subsets M, A, B, and C. We say $A \perp_w B \,|\, C$ if there is no active alternating path between A and B with respect to M in the partial-ancestor graph $G_{\text{anc}}^{V.M \cup A \cup B}$.

Figure 1.2(a) illustrates a DAG in eight nodes, $C = \{i, j\}$ and $M = \{h, l, q\}$. We also suppose that $A = \{k\}$ and $B = \{p, r\}$, therefore, $M \cup A \cup B = \{k, h, l, p, q, r\}$.

From Figure 1.2(b) we conclude that A and B are not w-separated by C since in the partial-ancestor graph $\langle p, i, l, k\rangle$ is an active alternating path with respect to M between A and B.

Global and pairwise Markov properties. For a graph G we say that an independence model $\mathcal{J}^G$ *satisfies the global Markov property with respect to* $\mathcal{S}$ if $A \perp_{\mathcal{S}} B \,|\, C$ implies $\langle A, B \,|\, C\rangle \in \mathcal{J}^G$.

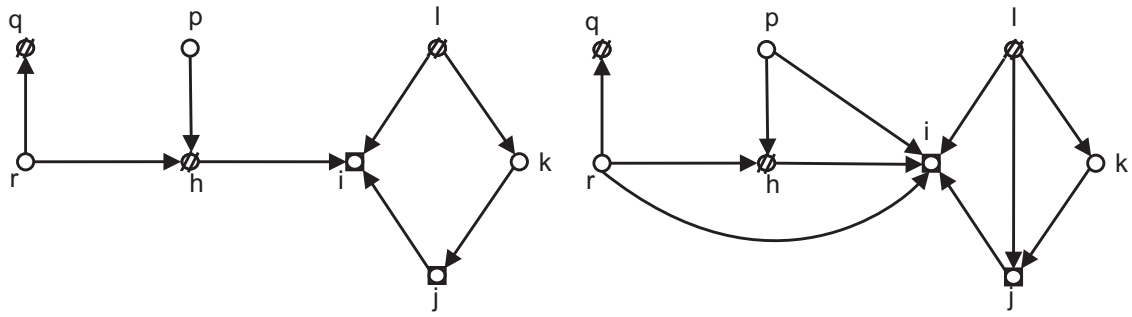


Figure 1.2: (a) A directed acyclic graph G with eight nodes, $\not\oslash \in M$ and $\boxed{\bigcirc} \in C$. (b) The partial-ancestor graph $G_{\text{anc}}^{V.M \cup A \cup B}$ generated from G.

Let h be a function from $\{i, j\}$ to $V \setminus \{i, j\}$. We also say that $\mathcal{J}^G$ *satisfies a pairwise Markov property with respect to* h if $i \not\sim j$ implies $\langle i, j \,|\, C \rangle \in \mathcal{J}^G$, for $C = h(i, j)$.

An example of global and pairwise Markov properties for DAGs. Applying the w-separation criterion on graph G in Figure 1.3 implies it only holds that $i \perp_w l \,|\, j$, $i \perp_w k \,|\, \varnothing$, $k \perp_w l \,|\, j$, $i \perp_w l \,|\, \{j, k\}$, $k \perp_w l \,|\, \{j, i\}$, and $\{i, k\} \perp_w l \,|\, j$.

For brevity if $\langle i, j \,|\, C \rangle \in \mathcal{J}^G$ then we assume $\langle j, i \,|\, C \rangle \in \mathcal{J}^G$ without writing it as a member of the set. It holds that $\mathcal{J}_1^G = \{\langle i, l \,|\, j \rangle, \langle i, k \,|\, \varnothing \rangle, \langle k, l \,|\, j \rangle, \langle i, l \,|\, \{j, k\} \rangle, \langle k, l \,|\, \{j, i\} \rangle, \langle \{i, k\}, l \,|\, j \rangle\}, \langle k, l \,|\, i \rangle\}$ satisfies the global Markov property with respect to $\perp_w$ (notice that even k is not w-separated from l by i) whereas $\mathcal{J}_2^G = \{\langle i, l \,|\, j \rangle, \langle i, k \,|\, \varnothing \rangle, \langle k, l \,|\, j \rangle, \langle i, l \,|\, \{j, k\} \rangle, \langle k, l \,|\, \{j, i\} \rangle, \langle k, l \,|\, i \rangle\}$ does not satisfy the global Markov property since $\langle \{i, k\}, l \,|\, j \rangle \notin \mathcal{J}_2^G$.

Let $h(x, y) = \text{an}(x) \cup \text{an}(y) \setminus \{x, y\}$. Let also $C_1 = h(i, l) = \{j, k\}$, $C_2 = h(i, k) = \varnothing$, and $C_3 = h(l, k) = \{i, j\}$. Independence models $\mathcal{J}_1^G$ and $\mathcal{J}_2^G$ both satisfy the pairwise Markov property since these contain $\langle i, l \,|\, C_1 \rangle$, $\langle i, k \,|\, C_2 \rangle$, and $\langle l, k \,|\, C_3 \rangle$.

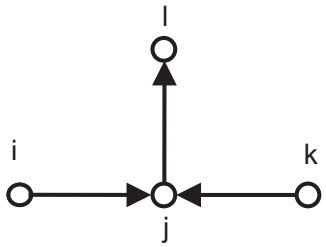


Figure 1.3: A directed acyclic graph G for which, e.g., $\{i,k\} \perp_w l \,|\, j$.

Compositional graphoid. Here the separation criteria whose induced independence models satisfy the (compositional) graphoid axioms are of interest to us, since otherwise, the graphical approaches cannot describe the independence model; see Pearl and Paz [1987]. An independence model $\mathcal{J}$ over a set V is a *semigraphoid* if, for disjoint subsets A,B,C, and D of V, it satisfies the four following properties:

1. $\langle A, B \,|\, C\rangle \in \mathcal{J}$ if and only if $\langle B, A \,|\, C\rangle \in \mathcal{J}$ (*symmetry*);
2. if $\langle A, B \cup D \,|\, C\rangle \in \mathcal{J}$ then $\langle A, B \,|\, C\rangle \in \mathcal{J}$ and $\langle A, D \,|\, C\rangle \in \mathcal{J}$ (*decomposition*);
3. if $\langle A, B \cup D \,|\, C\rangle \in \mathcal{J}$ then $\langle A, B \,|\, C \cup D\rangle \in \mathcal{J}$ and $\langle A, D \,|\, C \cup B\rangle \in \mathcal{J}$ (*weak union*);
4. $\langle A, B \,|\, C \cup D\rangle \in \mathcal{J}$ and $\langle A, D \,|\, C\rangle \in \mathcal{J}$ if and only if $\langle A, B \cup D \,|\, C\rangle \in \mathcal{J}$ (*contraction*).

A semigraphoid for which the reverse implication of the weak union property holds is said to be a *graphoid*, that is for disjoint subsets A,B,C, and D of V:

5. if $\langle A, B \,|\, C \cup D\rangle \in \mathcal{J}$ and $\langle A, D \,|\, C \cup B\rangle \in \mathcal{J}$ then $\langle A, B \cup D \,|\, C\rangle \in \mathcal{J}$ (*intersection*).

Furthermore, a graphoid or semigraphoid for which the reverse implication of the decomposition property holds is said to be *compositional*, that is for disjoint subsets A,B,C, and D of V:

6. if $\langle A, B \,|\, C\rangle \in \mathcal{J}$ and $\langle A, D \,|\, C\rangle \in \mathcal{J}$ then $\langle A, B \cup D \,|\, C\rangle \in \mathcal{J}$ (*composition*).

1.3.2 Probabilistic conditional independence models

In this section we relate independence models and separation criteria to the concept of probabilistic conditional independence. First, we give basic definitions of conditional independence in probability.

Definition of probabilistic conditional independence. If X, Y, and Z are discrete random variables with a joint distribution P we then say that X is *conditionally independent* of Y given Z, and use the notation, due to Dawid [1979], $X \perp\!\!\!\perp Y \,|\, Z$ if

$$P(X = x, Y = y \,|\, Z = z) = P(X = x \,|\, Z = z)P(Y = y \,|\, Z = z),$$

where $P(X = x \,|\, Y = y) = P(X = x, Y = y)/P(Y = y)$ and the equation holds for all z with $P(Z = z) > 0$. Furthermore, X is *independent* of Y, $X \perp\!\!\!\perp Y$ if and only if $X \perp\!\!\!\perp Y \,|\, Z$ for a trivial Z, i.e. $Z = \varnothing$. In the continuous case if $f_{XY}(x, y)$ is the joint density function of X and Y then the *conditional density* of Y given X is defined to be

$$f_{Y \,|\, X}(y \,|\, x) = \frac{f_{XY}(x, y)}{f_X(x)},$$

if $0 < f_X(x) < \infty$, and 0 otherwise [Rice, 2007], where $f_X(x)$ is known as the *marginal density* of X and defined as

$$f_X(x) = \int_{-\infty}^{\infty} f_{XY}(x, y)dy.$$

Then, if the three variables admit a joint density $f_{XYZ}(x, y, z)$ then we write $X \perp\!\!\!\perp Y \,|\, Z$ if

$$f_{XY \,|\, Z}(x, y \,|\, z) = f_{X \,|\, Z}(x \,|\, z) f_{Y \,|\, Z}(y \,|\, z),$$

where this equation is to hold for all z with $f_Z(z) > 0$. For more on properties of conditional independence; see Lauritzen [1996], Studeny [2005].

Probabilistic conditional independence and independence models. The most commonly interesting independence models are induced by probability distributions. Consider a set V and a collection of random variables $(X_\alpha)_{\alpha\in V}$ with joint density f_V. By letting $X_A = (X_v)_{v\in A}$ for each subset A of V, we then use the short notation $A \perp\!\!\!\perp B \,|\, C$ for $X_A \perp\!\!\!\perp X_B \,|\, X_C$ and disjoint subsets A, B, and C of V. There are two approaches to associate the probabilistic conditional independence and independence models.

The first approach is for a given a collection of random variables. From a given collection of random variables $(X_\alpha)_{\alpha\in V}$ with a probability distribution P, one can induce an independence model $\mathcal{J}(P)$ by demanding

$$\text{if } A \perp\!\!\!\perp B \,|\, C \text{ then } \langle A, B \,|\, C\rangle \in \mathcal{J}(P).$$

The other approach is for a given independence model $\mathcal{J}$. In this case a probability distribution P is called *faithful* with respect to $\mathcal{J}$ if, for random vectors X_A, X_B, and X_C with probability distribution P,

$$A \perp\!\!\!\perp B \,|\, C \text{ if and only if } \langle A, B \,|\, C\rangle \in \mathcal{J}.$$

We say that $\mathcal{J}$ is *probabilistic* if there is a distribution P that is faithful to $\mathcal{J}$. Notice that $\mathcal{J}(P)$ is obviously probabilistic.

Factorisation for DAGs. Now we relate the probabilistic conditional independence statement $X \perp\!\!\!\perp Y \,|\, Z$ to the separation criterion $X \perp Y \,|\, Z$ in the case of DAGs. For some other types of graphs, almost the same procedure applies.

Consider a directed acyclic graph $G = (V, E)$ and a collection of random variables $(X_\alpha)_{\alpha\in V}$ with joint density f_V. It is said that the joint distribution f_V *factorises* over the directed acyclic graph G if

$$f_V(x) = \prod_{v\in V} f_V(x_v \,|\, x_{\mathrm{pa}(v)}).$$

Geiger et al. [1990] (for another separation criterion which is equivalent to $\perp_w$; see Proposition 2.4) showed that for disjoint subsets A, B, and C of V, if the joint distribution f_V factorises over the directed acyclic graph $G = (V, E)$ then

$$A \perp_w B \,|\, C \;\text{ in } G \Rightarrow A \perp\!\!\!\perp B \,|\, C.$$

This can be translated as "the factorisation property implies the global Markov property for DAGs". Geiger et al. [1990] also showed that if A is not separated from B given C then there exists a joint distribution f such that f factorises over G and $A \perp\!\!\!\perp B \,|\, C$ does not hold.

For example, the factorisation of the joint distribution $f_{X_1,\ldots,X_8}(x_1, \ldots, x_8)$ over the graph in Figure 1.2 is

$$f(x_1, \ldots, x_8) = f(x_1 \,|\, x_2, x_4, x_5) f(x_2 \,|\, x_3) f(x_3 \,|\, x_5) f(x_4 \,|\, x_6, x_8)$$

$$f(x_5) f(x_6) f(x_7 \,|\, x_8) f(x_8),$$

where numbers $(1, \ldots, 8)$ correspond to nodes (i, j, k, h, l, p, q, r). By what we discussed in the last subsection and since the set $A = \{3\}$ is not w-separated from B=$\{6,8\}$ given $C = \{1, 2\}$, we conclude that $A \perp\!\!\!\perp B \,|\, C$ is not implied over the DAG in Figure 1.2.

Probabilistic conditional independence and compositional graphoids. Probabilistic independence models are semigraphoids [Pearl, 1988], whereas the converse is not necessarily true; see Studeny [1989].

Intersection and composition properties, however, do not necessarily hold in arbitrary probabilistic independence models. It is known that if the joint density of all variables is strictly positive then the intersection property also holds, and thus, the induced independence model is graphoid; for the proof see Proposition 3.1 in Lauritzen [1996].

There are, however, few well-studied families of distributions for which the com-

position property always holds. If the distribution P is a regular multivariate Gaussian distribution, $\mathcal{J}(P)$ is a compositional graphoid. This follows from the fact that for such a distribution

$$A \perp\!\!\!\perp B \,|\, C \iff k^{\alpha\beta}_{A\cup B\cup C} = 0 \text{ for all } \alpha \in A, \beta \in B,$$

where $k^{\alpha\beta}_{A\cup B\cup C}$ is the $\alpha\beta$ entry in the concentration matrix of the distribution of $X_{A\cup B\cup C}$. The family of symmetric binary variables is another example of such families; see Wermuth et al. [2009].

As we shall show in the next chapter, any independence model generated by an LMG is a compositional graphoid. Hence it follows that if a probabilistic independence model is faithful to an LMG, it is itself compositional. The statement in Section 1.6 of Wermuth [2011] saying that distributions *generated over a parent graph* satisfy the composition property reflects this fact in the case where the LMG is a DAG.

Instead of stating that probabilistic independence models satisfy the compositional graphoid axioms, we usually say that the probability distributions satisfy the axioms.

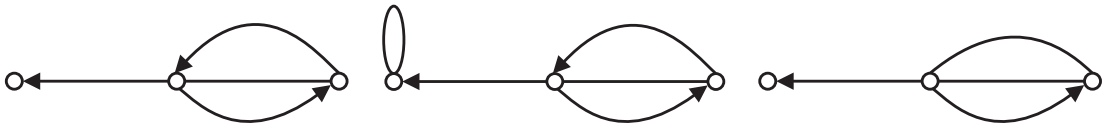
Chapter 2

Independence structures for mixed graphs

2.1 Introduction to loopless mixed graphs

Loopless mixed graphs. A *mixed graph* is a graph containing three types of edges denoted by arrows, arcs (two-headed arrows), and lines (full lines). Mixed graphs may have multiple edges of different types but do not have multiple edges of the same type. We do not distinguish between $i \multimap\!\!\multimap j$ and $j \multimap\!\!\multimap i$ or $i \prec\!\!\succ j$ and $j \prec\!\!\succ i$, but we do distinguish between $j \multimap\!\!\succ i$ and $i \multimap\!\!\succ j$. Thus there are up to four edges as a multiple edge between any two nodes. A *loopless mixed graph* (LMG) is a mixed graph that does not contain any loops (a loop may be formed by a line, arrow, or arc). Figure 2.1 illustrates examples of LMGs as well as mixed and non-mixed graphs. Graph H_2 is not an LMG because of the existence of a line loop, and H_3 is not a mixed graph because of a multiple edge consisting of two lines.

Goal of the chapter. In this chapter we introduce and discuss some special subclasses of LMGs and their associated independence models, many of which have been used in statistics and graphical models before. All these classes of



(a) (b) (c)

Figure 2.1: (a) An LMG H_1. (b) A mixed graph H_2 that is not an LMG. (c) A graph H_3 that is not a mixed graph.

graphs can be defined by providing subgraphs such that graphs that contain these should be excluded from the class. The largest subclass is the class of ribbonless graphs (used for MC graph models [Koster, 2002]), which contains all other classes as a subclass. Other classes are acyclic armorless graphs (used for summary graph models [Wermuth, 2011]), ancestral graphs [Richardson and Spirtes, 2002], acyclic directed mixed graphs [Spirtes et al., 1997], bidirected chain graphs (used for multivariate regression chain graph models [Cox and Wermuth, 1993]), undirected chain graphs, undirected graphs (used for concentration graph models), bidirected graphs (used for covariance graph models), and DAGs.

The common feature of ribbonless graphs and its subclasses is that these all entail independence models using the same separation criterion, called m-separation. These subclasses seem to cover all graphical models and their independence interpretations that have been used in the literature except some independence interpretations of chain graphs; see Section 3. The m-separation criterion induces independence models that are compositional graphoid. Some of these graphs are maximal in the sense that every missing edge is related to an independence statement. For the ribbonless graphs that are not maximal, we generate maximal graphs of the same type that induce the same independence model by adding edges to the graph. For maximal ribbonless graphs, a pairwise Markov property that is satisfied by the independence model can be defined and, for an independence model that satisfies the compositional graphoid axioms, the defined

pairwise Markov property is proven to be equivalent to the global Markov property associated with the m-separation criterion.

The class of RGs does not have a known parametrisation of the corresponding set of distributions even in the Gaussian case. However, a motivation for defining such a class is to use and simplify their properties for the many known subclasses.

Structure of the chapter. In this section we give some graph theoretical definitions and notations for mixed graphs. We also define the m-separation criterion and prove that this is a compositional graphoid.

In Section 2, we introduce a type of LMGs called ribbonless graphs. In addition we define anterior graphs and prove that RGs induce the same independence model as their anterior graphs by using m-separation. We also show that even though ribbonless graphs are not the same as the known class of MC graphs, by a simple modification on MC graphs, the ribbonless graphs will induce the same independence model by the m-separation criterion.

In Section 3, we define each special type of LMG independently and show how these are related to known graphical models. We also investigate whether the m-separation criterion can be further simplified on them.

In Section 4, we show that ribbonless graphs are not necessarily maximal, and we provide conditions under which a ribbonless graph is maximal and an algorithm to generate a maximal graph of the same type that induces the same independence model. We also show which subclasses of LMGs are maximal and which are not.

In section 5, we define a pairwise Markov property for maximal ribbonless graphs. We prove that in the case of a compositional graphoid, the defined pairwise Markov property is equivalent to the global Markov property induced by the m-separation criterion. This leads to an important corollary regarding the equiv-

alence of Markov properties for probabilistic independence models.

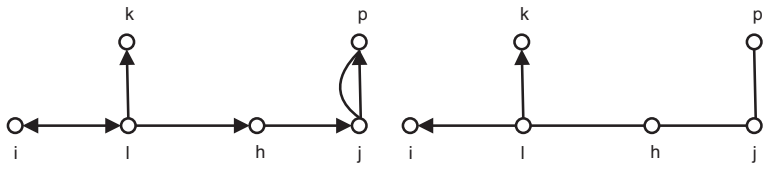
Some basic definitions for mixed graphs. For a mixed graph H, we redefine some of the same terminology introduced before for directed and undirected graphs so that they conform with the definitions introduced for $H[\twoheadrightarrow]$, i.e. the subgraphs induced by arrows (for directed graphs) and $H[\,\text{—}]$, i.e. the subgraph induced by lines (for undirected graphs). In particular, for determining the ancestors, a direction-preserving path in a mixed graph is also direction-preserving in $H[\twoheadrightarrow]$, i.e. it only consists of arrows in one direction. Notice also that, for mixed graphs, we use the notation $i \sim j$ when there is an edge (of any type) between i and j. In addition, we say that i is a *neighbour* of j if these are endpoints of a line, and i is a parent of j if there is an arrow from i to j. We also define that i is a *spouse* of j if these are endpoints of an arc. We use the notations $\mathrm{ne}(j)$, $\mathrm{pa}(j)$, and $\mathrm{sp}(j)$ for the set of all neighbours, parents, and spouses of j respectively. In the cases of $i \rightarrow j$ or $i \leftrightarrow j$ we say that there is *an arrowhead pointing to (at)* j. In a mixed graph the inner node of three V-configurations $i \rightarrow t \leftarrow j$, $i \leftrightarrow t \leftarrow j$, and $i \leftrightarrow t \leftrightarrow j$ is a *collider* and the inner node of all other V-configurations is a *non-collider*. In fact, the collider node is an extension of the collision node for mixed graphs. We also call a V-configuration with collider or non-collider inner node *collider* or *non-collider V-configuration* respectively.

Two paths π_1 and π_2 (including V-configurations or edges) between i and j are called *endpoint-identical* if there is an arrowhead pointing to i in π_1 if and only if there is an arrowhead pointing to i in π_2 and similarly for j. For example, the paths $i \rightarrow j$, $i \text{——} k \leftrightarrow j$, and $i \rightarrow k \leftarrow l \leftrightarrow j$ are all endpoint-identical as they have an arrowhead pointing to j but no arrowhead pointing to i on the paths.

Anterior graphs and sets. The *anterior graph* of a loopless mixed graph G, denoted by G^*, is the graph obtained from G by repeatedly removing arrowheads pointing to nodes that are the endpoints of a line, i.e. by obtaining $\multimap\!\circ\!\multimap$ and $\leftarrow\!\circ\!\multimap$ from $\rightarrow\!\circ\!\multimap$ and $\leftrightarrow\!\circ\!\multimap$ respectively, and considering only one edge of the same type between each pair of nodes. A path $\langle i = i_0, i_1, \ldots, i_n = j\rangle$ between i and j in G^* is an *anterior path from* i to j if it has the form $i \multimap i_1 \multimap \ldots \multimap i_m \rightarrow i_{m+1} \rightarrow \ldots \rightarrow j$.

We call i an *anterior* of j if there is an anterior path from i to j in G^*. We use the notation $\mathrm{ant}(i)$ for the set of all anteriors of i. Notice that, since, as we shall see, ancestral graphs have no arrowheads pointing to lines, we have $G = G^*$ for an ancestral graph. Thus our definition of anterior extends the notion of anterior used in Richardson and Spirtes [2002] for ancestral graphs. However, it is different from and inconsistent with the definition of anteriors in Frydenberg [1990].

For example, in the graph G in Fig. 2.2(a), $\mathrm{ant}(i) = \{l, h, j, p\}$ and $\mathrm{ant}(p) = \{l, h, j\}$. This can be seen by looking at the anterior paths $\langle p, j, h, l, i\rangle$ from p to i and $\langle l, h, j, p\rangle$ from l to p (as well as from p tow l) in Fig. 2.2(b).



(a) (b)

Figure 2.2: (a) A mixed graph G. (b) The anterior graph G^* of G.

A cycle is called an *anterior cycle* if the nodes on the cycle are anteriors of themselves via the cycle.

We first show that transitivity holds for anteriors.

Lemma 2.1. *For any loopless mixed graph it holds that if* $i \in \mathrm{ant}(j)$ *and* $j \in$

$\mathrm{ant}(k)$ *then* $i \in \mathrm{ant}(k)$.

Proof. If $i \in \mathrm{ant}(j)$ and $j \in \mathrm{ant}(k)$ then there are anterior paths π_1 from i to j and π_2 from j to k and, since no arrowhead meets a line in G^*, their combination $\pi_1 \circ \pi_2$ is an anterior path from i to j. □

Here we also introduce a Lemma that is widely used in the proofs of this chapter.

Lemma 2.2. *If* $i \in \mathrm{ant}(j) \setminus \mathrm{an}(j)$ *then either* i *or a descendant of* i *is an endpoint of a line.*

Proof. Suppose that $\langle G = G_0, G_1, \ldots, G_n = G^* \rangle$ is a sequence of graphs, where each graph has been generated by removing one arrowhead pointing to a full line from the previous graph starting from G. We prove the result by reverse induction along this sequence. Notice first that the anterior sets are the same in all these graphs.

For the base, the result holds for G_n. This is because there is a path of form $i \multimap \ldots \multimap \circ\!\!\rightarrow \ldots \rightarrow j$ in G_n and, since $i \notin \mathrm{an}(j)$, the path consists of at least one full line.

For the inductive step we show that if the result holds for G_m then the result also holds for G_{m-1}. Suppose that an arrowhead pointing to a node h on the lh-edge has been removed at this step. We also know that, in G_{m-1}, $i \notin \mathrm{an}(j)$. We now deal with two cases:

Case I. If, in G_m, $i \in \mathrm{an}(j)$ then the lh-edge is an arc turning into an arrow that makes a direction-preserving path from i to j. In this case h is either i or a descendant of i and an endpoint of a line.

Case II. If, in G_m, $i \notin \mathrm{an}(j)$ then by the induction hypothesis, in G_m, there is a node k that is either i or a descendant of i and that is an endpoint of a line. Here we deal with two cases again: 1) Suppose that the line adjacent to k is an

arrow from h to l in G_{m-1}. If $h = k$ then in G_{m-1}, k is still i or a descendant of i and is a parent of an endpoint of a line. If $h = l$ then in G_{m-1}, k is still i or a descendant of i and is itself an endpoint of a line. Therefore i or a descendant of i is an endpoint of a line. 2) If in G_{m-1} there is an lh-arc turning into an arrow that makes a direction-preserving path from i to k then h, which is an endpoint of a line, is i or a descendant of i. □

The m-separation criterion. Here we define a separation criterion for LMGs, even though from Section 2 onward in this chapter we only focus on ribbonless graphs. We use this criterion to induce independencies on LMGs and its subclasses defined in Section 3. To define the criterion, we first define a special type of path called m-connecting path:

Let C be a subset of the node set of an LMG. A path is *m-connecting* given C if all its collider nodes are in $C \cup \text{an}(C)$ and all its non-collider nodes are outside C. For two other disjoint subsets of the node set A and B, we say that C m-separates A and B if there is no m-connecting path between A and B given C. In this case we use the notation $A \perp_m B \,|\, C$. Notice that the m-separation criterion induces an independence model $\mathcal{J}_m(G)$ on G by $A \perp_m B \,|\, C \iff \langle A, B \,|\, C\rangle \in \mathcal{J}_m(G)$.

The m-separation criterion for LMGs is the same as the separation criterion defined in Richardson and Spirtes [2002]. This itself is an extension of the d-separation criterion, as introduced by Pearl [1988]. Clearly, m-separation is also an extension of simple separation in an undirected graph, as then all edges are lines.

For example, in graph H in Figure 2.3 it holds that $h \in \text{an}(l)$ and, thus, $\langle i, h, j\rangle$ is an m-connecting path given l. Therefore, $\langle i, j \,|\, l\rangle \notin \mathcal{J}_m(H)$.

The m-separation criterion and graphoid axioms. Here we prove that $\mathcal{J}_m$ satisfies the compositional graphoid axioms, defined in Chapter 1.

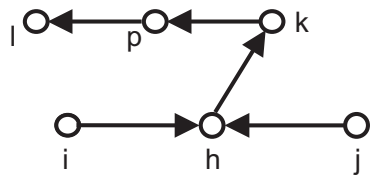


Figure 2.3: A loopless mixed graph H for which, e.g., $\langle i, j \,|\, l\rangle \notin \mathcal{J}_m(H)$.

Theorem 2.1. *Let H be a loopless mixed graph. The independence model $\mathcal{J}_m(H)$ is then a compositional graphoid.*

Proof. For $H = (N, F)$ and disjoint subsets A, B, C, and D of N, we prove that $\perp_m$ satisfies the six compositional graphoid axioms:

1) *Symmetry:* If $A \perp_m B \,|\, C$ then $B \perp_m A \,|\, C$: If there is no m-connecting path between A and B given C then there is no m-connecting path between B and A given C.

2) *Decomposition:* If $A \perp_m (B \cup D) \,|\, C$ then $A \perp_m D \,|\, C$: If there is no m-connecting path between A and $B \cup D$ given C then there is no m-connecting path between A and $D \subseteq (B \cup D)$ given C.

3) *Weak union:* If $A \perp_m (B \cup D) \,|\, C$ then $A \perp_m B \,|\, (C \cup D)$: From (2) we know that $A \perp_m D \,|\, C$ and $A \perp_m B \,|\, C$. Suppose, for contradiction, that there exist m-connecting paths between A and B given $C \cup D$. Consider a shortest path of this type and call it π. If there is no inner collider node on π then there is an m-connecting path between A and B given C, a contradiction. On π all collider nodes are in $(C \cup D) \cup \mathrm{an}(C \cup D)$. If all collider nodes are in $C \cup \mathrm{an}(C)$ then there is an m-connecting path between A and B given C, again a contradiction. Hence consider the closest collider node $i \in (D \cup \mathrm{an}(D)) \setminus (C \cup \mathrm{an}(C))$ to A on π. Now since the nodes between A and i are not in $B \cup D$, there is an m-connecting path between A and i given C. If $i \in D$ then this is obviously a contradiction, otherwise there is an m-connecting path between A and a node $k \in D$, such that

$i \in \text{an}(k)$, given C, a contradiction again. Therefore, there is no m-connecting path between A and B given $C \cup D$.

4) *Contraction:* If $A \perp_m B \,|\, C$ and $A \perp_m D \,|\, (B \cup C)$ then $A \perp_m (B \cup D) \,|\, C$: Suppose, for contradiction, that there exists an m-connecting path between A and $B \cup D$ given C. Consider a shortest path of this type and call it π. The path π is either between A and B or between A and D. The path π being between A and B contradicts $A \perp_m B \,|\, C$. Therefore, π is between A and D. In addition, since all inner collider nodes on π are in $C \cup \text{an}(C)$ and because $A \perp_m D \,|\, (B \cup C)$, an inner non-collider node should be in B. This contradicts the fact that π is a shortest m-connecting path between A and $B \cup D$ given C.

5) *Intersection:* If $A \perp_m B \,|\, (C \cup D)$ and $A \perp_m D \,|\, (C \cup B)$ then $A \perp_m (B \cup D) \,|\, C$: Suppose, for contradiction, that there exists an m-connecting path between A and $B \cup D$ given C. Consider a shortest path of this type and call it π. The path π is either between A and B or between A and D. Because of symmetry between B and D in the formulation it is enough to suppose that π is between A and B. Since all inner collider nodes on π are in $C \cup \text{an}(C)$ and because $A \perp_m B \,|\, (C \cup D)$, an inner non-collider node should be in D. This contradicts the fact that π is a shortest m-connecting path between A and $B \cup D$ given C.

6) *Composition:* If $A \perp_m B \,|\, C$ and $A \perp_m D \,|\, C$ then $A \perp_m (B \cup D) \,|\, C$: Suppose, for contradiction, that there exist m-connecting paths between A and $B \cup D$ given C. Consider a path of this type and call it π. Path π is either between A and B or between A and D. Because of symmetry between B and D in the formula it is enough to suppose that π is between A and B. But this contradicts $A \perp_m B \,|\, C$. □

The following seemingly trivial corollary of the theorem , mostly without being mentioned explicitly, is very commonly used in the literature and in the proofs of this thesis:

Corollary 2.1. *Let H be a loopless mixed graph and A, B, and C be disjoint subsets of its node set. It holds that $A \perp_m B \mid C$ if and only if $i \perp_m j \mid C$ for every nodes $i \in A$ and $j \in B$.*

Proof. The result follows from the fact that $\perp_m$ satisfies the decomposition and the composition properties. □

2.2 Ribbonless graphs and their relationship to MC graphs

Ribbonless graphs. The largest subclass of LMGs defined in this chapter is the class of ribbonless graphs.

A *ribbon* is a collider V-configuration $\langle h, i, j \rangle$ such that

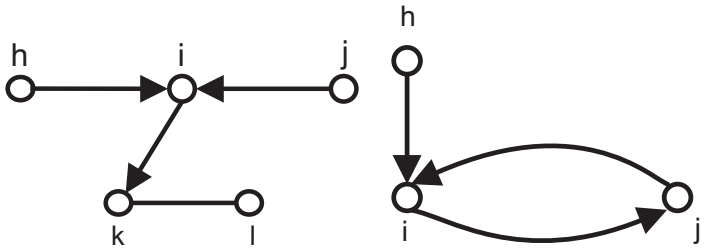
1. there is no endpoint-identical edge between h and j, i.e. there is no hj-arc in the case of $h \prec\!\!\succ i \prec\!\!\succ j$; there is no hj-line in the case of $h \longrightarrow i \prec\!\!\!-\!\!\!- j$; and there is no arrow from h to j in the case of $h \longrightarrow i \prec\!\!\succ j$;
2. i or a descendant of i is an endpoint of a line or a direction-preserving cycle.

If i or a descendant of i is an endpoint of a line then we call the ribbon to be of type (i) and if these are on a direction-preserving cycle then we call the ribbon to be of type (ii). Notice that a ribbon might be of both types (i) and (ii). A *ribbonless graph* (RG) is an LMG that does not contain ribbons as induced subgraphs.

Notice that similar to many graph-theoretical definitions in graphical models, ribbon is a non-local concept, meaning that it is not apparent whether a V-configuration is a ribbon by just looking at the V-configuration, and not the rest of the graph.

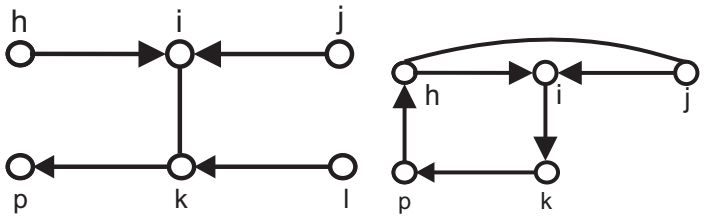
Figure 2.4 illustrates a ribbon $\langle h, i, j \rangle$ of type (i) and the simplest ribbon of type (ii). Figure 2.5(a) illustrates a graph containing a ribbon $\langle h, i, j \rangle$. Figure 2.5(b)

illustrates a ribbonless graph. Notice that $\langle h, i, j \rangle$ is not here a ribbon since there is a line between h and j.



(a) (b)

Figure 2.4: (a) A ribbon $\langle h, i, j \rangle$ of type (i), where $\text{ne}(i) = \varnothing$. (b) The simplest ribbon $\langle h, i, j \rangle$ of type (ii).



(a) (b)

Figure 2.5: (a) A graph that is not ribbonless. (b) A ribbonless graph.

We proceed to establish that ribbonless graphs yield identical independence models to their anterior graphs and need the following lemma.

Lemma 2.3. *Let G be a ribbonless graph. If there is a collider V-configuration $\langle i, j, k \rangle$ in G that is non-collider in G^*, then G has an ik-edge that is endpoint-identical to $\langle i, j, k \rangle$.*

Proof. Suppose that $\langle G = G_0, G_1, \ldots, G_n = G^* \rangle$ is a sequence of graphs, where each graph has been generated by removing one arrowhead pointing to a full line from the previous graph starting from G.

Consider the first intermediate graph G_{p+1} where $\langle i, j, k \rangle$ turns into a non-collider

V-configuration. First we prove by reverse induction the claim that, for each $0 \leq q \leq p$, j or a descendant of j is an endpoint of a line in G_q:

In G_p, the node j is obviously an endpoint of a line and the claim holds. Thus we assume that the claim holds for G_{q+1}, i.e. j or a descendant of j is an endpoint of an hl-line in G_{q+1}. In G_q, it is easy to observe that if the hl-line is an arrow pointing to another line or if an arrow on the direction-preserving path that points to the hl-line is an arc then j or a descendant of j is still an endpoint of a line. Therefore, the claim holds in G_q. Therefore, by reverse induction, this claim holds in G.

Now $\langle i, j, k\rangle$ is a ribbon of type (i) unless there is an endpoint-identical ik-edge to $\langle i, j, k\rangle$, which is the case since G is ribbonless. □

Proposition 2.1. *For a ribbonless graph G, it holds that $\mathcal{J}_m(G) = \mathcal{J}_m(G^*)$, i.e. G and G^* are Markov equivalent.*

Proof. It is enough to prove that there is an m-connecting path between i and j given C in G if and only if there is an m-connecting path between i and j given C in G^*.

Suppose that there is an m-connecting path between i and j given C in G. All non-colliders on the path in G are preserved in G^*. In addition, by Lemma 2.3, a collider V-configuration $\langle i, j, k\rangle$ becomes non-collider if there is an endpoint-identical ik-edge to $\langle i, j, k\rangle$. In this case the ik-edge can be used instead of $\langle i, j, k\rangle$ to establish an m-connecting path in G^*.

Conversely, suppose that there is an m-connecting path between i and j given C in G^*. Collider V-configurations are collider V-configurations in G, and if a non-collider V-configuration $\langle i, j, k\rangle$ has been collider in G then, by Lemma 2.3, one can again use the ik-edge instead of $\langle i, j, k\rangle$.

Thus the only thing that remains to be proven is when a direction-preserving

path (pointing to a member of C) in G does not remain direction-preserving in G^* or vice versa.

In this case, suppose that there is a collider V-configuration $\langle i, j, k\rangle$ in both G and G^*, and $j \in \mathrm{an}(C)$ in one of the graphs. By the same argument as in Lemma 2.3, if the arrowhead of an arrow on the direction-preserving path in G is taken away or the arrowhead of an arc in G is taken away to generate a direction-preserving path in G^* then $\langle i, j, k\rangle$ is a ribbon unless there is an endpoint-identical ik-edge to $\langle i, j, k\rangle$. Hence again we can use the ik-edge instead of $\langle i, j, k\rangle$ for establishing an m-connecting path. □

Thus the absence of ribbons ensures that the Markov property is unchanged by forming the anterior graph G^*. In addition, independence models induced by m-separation in a ribbonless graph can be induced by marginalisation over and conditioning on a DAG-independence model. For details, see the next chapter, and in particular Proposition 3.9. This implies that independence models corresponding to RGs are probabilistic since the independence models corresponding to DAGs are probabilistic.

RGs and MC graphs. We use the m-separation criterion on RGs to induce an independence model. Here we discuss the relationship between RGs and the class of MC graphs, defined by Koster [2002]. In MC graphs a different separation criterion is used for inducing the independence model. We will show that even though these two classes of graphs and their independence interpretations are not the same, from an MC graph and by a minor modification one can generate an RG that induces the same independence model.

Definition of MC graphs. Koster [2002] defines *MC graphs* to be exactly the same as the class of mixed graphs, even though it is shown in this paper that among three types of loops only the existence of line loops is necessary for

the purpose of generating stable independence graphs under marginalisation over and conditioning on the node sets of DAGs. "M" and "C" in MC graph stand for marginalisation and conditioning respectively. Marginalisation and conditioning for DAGs will be discussed comprehensively in the next chapter.

In addition, by the method proposed to generate MC graphs after marginalisation and conditioning in Table 1 in Koster [2002], which is closely related to Table 3.1 in this thesis, some graphs cannot be generated as MC graphs after marginalisation and conditioning for DAGs (even though Koster has not precisely talked about the graphs that are generated by this algorithm). To be more precise, the graphs that are generated by this method are ribbonless or are those containing a ribbon $\langle i, k, j \rangle$ with k being an endpoint of or being an ancestor of a node that is an endpoint of a line loop. We call these ribbons *loop-ribbons* and all other types of ribbons *non-loop-ribbons*. The reason is that, if there is a line or a direction-preserving cycle in the MC graph generated by a DAG then in the DAG there is an arrow pointing to a node c that is conditioned on. (We do not go through the details of proof here, but it uses a similar result as Lemma 3.2.) This implies that when there is an arrow from node k pointing to c, a line loop on k is generated. Therefore, *MC graphs generated by DAGs* do not contain non-loop-ribbons.

A separation criterion on MC graphs. The separation criterion for MC graphs is different from the m-separation criterion, which we use in this thesis. In this criterion what we call in this thesis a walk is used instead of paths, even though in Koster [2002], for such an object, the term path was used. In addition, despite what we have defined before, here the endpoints of a V-configuration are not necessarily distinct.

In a graph $H = (N, F)$ define $C_\pi := \{j \in N : j \text{ occurs as a collider on } \pi\}$ and $N_\pi := \{j \in N : j \text{ occurs as a non-collider on } \pi\}$. Suppose that π is a walk between nodes i and j and consider a subset C of N that does not contain i and j.

Then π is *open given* C if $C_\pi \subseteq C$ and $N_\pi \subseteq N \setminus C$. If the walk is not open given C then it is *blocked by* C. We say $A \perp_k B \,|\, C$ if all walks between A and B are blocked by C.

The difference between this separation criterion and m-separation is that a collider (a member of C_π) should be in C and cannot be in its ancestors. As we shall see the repetition of the edge makes this criterion equivalent to m-separation. For example, in the DAG in Figure 2.6 we conclude that k and h are not k-separated given i because of the walk $\pi = \langle h, j, i, j, k \rangle$. Here $C_\pi = i$ and $N_\pi = j$. therefore, π is open given i. By the m-separation criterion we can make the same conclusion on the separation of k and h given i by considering the m-connecting $\langle k, j, h \rangle$ path since $j \in \text{an}(i)$.

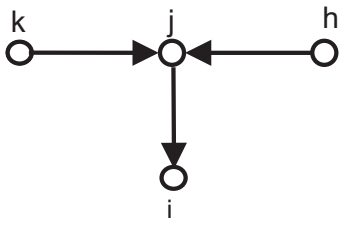


Figure 2.6: A DAG for which k and h are not k-separated given i.

An algorithm for generating RGs from MC graphs. Here we present a simple method to transform MC graphs into RGs. We also show that, by applying the m-separation criterion to the generated graph, the same independence model is yielded.

Algorithm 2.1. *(Generating an RG from an MC graph H generated by a DAG) Start from H.*

1. *For a loop-ribbon $\langle i, j, k \rangle$ generate an endpoint-identical edge between i and k.*

2. *Remove all loops.*

Apply step 1 until no other edge can be generated before moving to step 2.

For example, in Figure 2.7 we illustrate how to apply Algorithm 2.1 to a given MC graph.

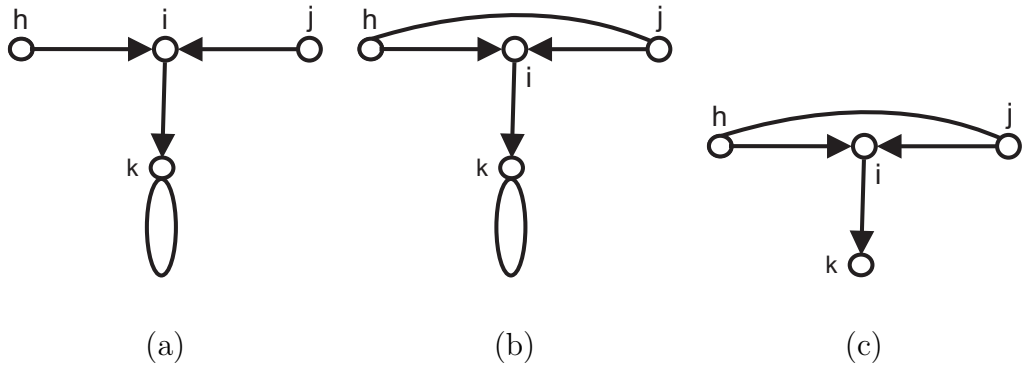


Figure 2.7: (a) An MC graph H. (b) The generated graph after applying step 1. (c) The generated RG after applying step 2

Lemma 2.4. *The choices of which current loop-ribbon to close next do not affect the final graph obtained by Algorithm 2.1.*

Proof. The result follows from the fact that adding an edge between endpoints of a loop-ribbon does not destroy other loop-ribbons in the graph, and it does not destroy the possible loop-ribbons that might be generated by later iterations of the algorithm unless it closes the loop-ribbon. Hence at each iteration of the algorithm (even with newly generated loop-ribbons) if there are two loop-ribbons then it does not matter which one to be chosen first. □

Proposition 2.2. *Algorithm 2.1 generates RGs.*

Proof. The generated graph is an LMG since by step 2 all loops are removed. Here we prove that the graph is ribbonless: First of all, step 1 removes all loop-ribbons from the graph. Thus it is enough to prove that there is no non-loop-ribbon in the generated graph. We know that the result is true in the input graph H since H is an MC graph generated by a DAG.

Consider one fixed order of applying the steps of Algorithm 2.1. Consider also the collection $\mathcal{C}$ of added edges by this algorithm. For any iteration G of the algorithm we say that a non-loop ribbon $\langle i, k, j\rangle$ is *durable* if the corresponding endpoint-identical ij-edge is not in $\mathcal{C}$. Now, we show by contradiction that no durable non-loop ribbon will be created.

Consider the first iteration of the algorithm where such a ribbon is created. Denote this by $\langle g, q, q_1\rangle$. We need to consider the following cases:

1) Suppose that the added ij-edge is not on the ribbon $\langle g, q, q_1\rangle$ but is an arrow from i to j on a direction-preserving cycle containing q or on a direction-preserving path from q to a node that is an endpoint of a line. Denote the loop-ribbon that generates the ij-edge by $\langle i, k, j\rangle$. We know that k or one of its descendants is an endpoint of a line loop, and the ik-edge is an arrow from i to k. This implies that a descendant of i is an endpoint of a line loop, which itself implies that a descendant of q is an endpoint of a line loop. Therefore, $\langle g, q, q_1\rangle$ is a loop-ribbon, not a non-loop-ribbon in G. Thus this case is not possible.

2) Suppose that the added ij-edge is not on the ribbon $\langle g, q, q_1\rangle$ but is a line such that there is a direction-preserving path from q to i. In this case again by the same argument as (1) we conclude that $\langle g, q, q_1\rangle$ is not a non-loop-ribbon, and the case is not possible.

3) Suppose that the gq-edge has been generated by a loop-ribbon $\langle g, t, q\rangle$. We observe that in the preceding iteration there must be a collider V-configuration $\langle t, q, q_1\rangle$ that is a non-loop ribbon. Thus, by the assumption, this ribbon is not durable, and the corresponding end-point identical tq_1-edge must be added during the algorithm. Once it is added $\langle g, t, q_1\rangle$ becomes a loop-ribbon. Therefore, an endpoint-identical gq_1-edge to $\langle g, t, q_1\rangle$ must be added that is also endpoint-identical to $\langle g, q, q_1\rangle$. This contradicts the assumption that $\langle g, q, q_1\rangle$ is durable.

□

Now we should prove that the m-separation criterion on the generated RG induces the same independence model as the independence model induced by the Koster's criterion on the original MC graph.

Proposition 2.3. *An MC graph H that is generated by a DAG with separation criterion $\perp_k$ and the RG generated from H by Algorithm 2.1 with separation criterion $\perp_m$ induce the same independence model.*

Proof. Denote the RG generated by Algorithm 2.1 from H by $\tilde{H}$. We prove that for an arbitrary subset C and two arbitrary nodes i and j, which are not in C, there is an open walk given C between i and j in H if and only if there is an m-connecting path given C between i and j in $\tilde{H}$.

($\Rightarrow$) Suppose that there is an open walk given C in H. Let H^+ denote the graph obtained by the algorithm from H before deleting the loops. Let π be an open ij-walk in H^+ given C which has as few repeated nodes as possible. We shall show that π has no repeated nodes, and so π is an m-connecting ij-path given C in H^+ and so in $\tilde{H}$.

If there is a repeated node on π then π should be of form $\langle i, \ldots, g, k, h = h_0, h_1, \ldots, h_n, k, l, \ldots, j\rangle$, where k is the closest node to i that appears on π for the second time. (Notice that some h_p, $0 \leq p \leq n$, may be the same nodes and in particular it is possible that $h = h_n$.) We denote $\langle k, h, \ldots, h_n, k\rangle$ by π'.

If there is no arrowhead pointing to k on the gk-edge or on the kl-edge then $k \notin C$ and, therefore, by skipping π', $\langle i, \ldots, g, k, l, \ldots, j\rangle$ is still open. Hence, in H^+ the V-configuration $\langle g, k, l\rangle$ can be on an m-connecting path. Thus suppose that there are arrowheads at k on gk- and kl-edges. If $k \in C$ then we can easily skip π' again.

Thus suppose that $k \notin C$. In this case we know that there is no arrowhead pointing to k on kh- and kh_n-edges. Therefore, we have the two following cases for the kh-edge: 1) It is a line. 2) It is an arrow from k to h, which implies that

$k \in \text{an}(h)$. In case (2) we proceed to the hh_1-edge. (Notice that it is possible that $k = h_1$.) Now we have the following cases: 1) The hh_1-edge is a line. 2) There is an arrowhead at h on the hh_1-edge, which implies that $h \in C$ and, therefore, $k \in \text{an}(C)$. 3) The hh_1-edge is an arrow from h to h_1, which implies that $k \in \text{an}(h_1)$, and we proceed to h_2. By induction we conclude that one of the following holds: a) k or a descendant of k is an endpoint of a line. b) k is on a direction-preserving cycle. c) $k \in \text{an}(C)$.

The first two cases imply that $\langle g, k, l \rangle$ is a ribbon, and hence a loop-ribbon. Therefore instead of this ribbon one can use the gl-edge, generated by step 1 of the algorithm to construct an m-connecting path in H^+. Notice that the gl-edge and $\langle g, k, l \rangle$ are endpoint-identical. If case (c) holds then $k \in \text{an}(C)$ also in H^+, and $\langle g, k, l \rangle$ can again be on an m-connecting path in H^+.

($\Leftarrow$) Suppose that there is an m-connecting path π_1 given C between i and j in $\tilde{H}$. Suppose also that $\langle H = H_0, H_1, \ldots, H_n = \tilde{H} \rangle$ is a sequence of graphs, where each graph has been generated by adding one edge to the previous graph starting from H. We prove by reverse induction along this sequence that there is an open walk given C between i and j in each graph, including H.

For the base, i.e. for $H_n = \tilde{H}$, if all collider nodes on π_1 are in C then π_1 is also an open walk given C. If a collider node k is in $\text{an}(h)$, where $h \in C$, then in $\langle i, \ldots, k, \ldots, h, \ldots, k, \ldots, j \rangle$ is an open walk given C, where the subpath $\langle k, \ldots, h, \ldots, k \rangle$ is a direction-preserving path from k to h and the rest has been copied from π_1.

Thus now suppose that there is an open walk π_2 given C in H_p. We prove the result for H_{p-1}. Suppose that at this stage the gl-edge has been generated by the algorithm. If this edge is not on π_2 then π_2 is obviously an open walk in H_{p-1}. Thus suppose that the gl-edge is on π_2.

In H_{p-1}, there exists an endpoint-identical loop-ribbon $\langle g, q, l\rangle$ to the gl-edge. Hence either q is an endpoint of a line loop or there exists a direction-preserving path π_3 from q to t such that t is an endpoint of a line loop. If there is no node in C on π_3 let $\hat{\pi}_3 = \pi_3$; and otherwise let $\hat{\pi}_3$ be the part of π_3 from q up to the first node in C. Now by replacing the gl-edge by $\langle g, q, q, l, \ldots, l\rangle$ or $\langle g, \hat{\pi}_3, \hat{\pi}_3^r, l\rangle$, where $\hat{\pi}_3^r$ is $\hat{\pi}_3$ in the reverse direction, we obtain an open walk in H_{p-1}. Now by reverse induction, the result follows. □

For example, for the graph H in Figure 2.7, the node h is not independent of j in either of the graphs because of the walk $\langle h, i, k, k, i, j\rangle$ in H and the hj-edge in the generated RG.

2.3 Subclasses of loopless mixed graphs

Acyclic armorless graphs and summary graphs. Here we define a subclass of LMGs, which has been recently used in statistics.

An *acyclic armorless graph* (AAG) is a mixed graph $H = (N, F)$ which contains no $\circ\!\!-\!\!\circ\!\!\prec\!\!\circ$ or $\circ\!\!-\!\!\circ\!\!\prec\!\!\succ\!\!\circ$ (*armor*) and no direction-preserving cycle. Notice that there are no multiple edges in AAGs except multiple edges consisting of an arrow and an arc. Figure 2.8 illustrates an AAG and an LMG that is not an AAG. (Because of two reasons: existence of an armor and existence of a double edge containing line and arrow.)

(a) (b)

Figure 2.8: (a) An AAG. (b) An LMG that is not an AAG.

AAGs coincide with the class of graphs defined by Wermuth [2011] called

summary graphs. Definition of AAG is the direct translation of the definition of summary graphs in Wermuth [2011]; see Definition 7 of Wermuth [2011]. Wermuth [2011] uses the notation of dashed line, ---, instead of arcs, but these have the same meanings in both texts.

Ancestral graphs. An *ancestral graph* (AG) is a simple mixed graph that has the following properties for all nodes i:

1. $i \notin \text{an}(\text{pa}(i) \cup \text{sp}(i))$;
2. if $\text{ne}(i) \neq \varnothing$ then $\text{pa}(i) \cup \text{sp}(i) = \varnothing$.

This means that there is no arrowhead pointing to a line and there is no direction-preserving cycle, and there is no so-called *bow*, i.e. an arc one of whose endpoint is the ancestor of the other endpoint, in the graph. Figure 2.9 illustrates examples of an AG and an AAG that is not ancestral.

(a) (b)

Figure 2.9: (a) An AG. (b) An AAG that is not ancestral.

AGs were originally defined and studied by Richardson and Spirtes [2002]. This class of graphs has been widely used in statistics and machine learning. Notice that in Richardson and Spirtes [2002] condition 1 of the definition was stated as $i \notin \text{ant}(\text{pa}(i) \cup \text{sp}(i))$. In addition, in this paper, the exposition of the definition of anteriors is different: i is said to be anterior to j if there is a path on which every edge is either of the form $k \text{——} l$, or $k \text{—→} l$ with k between i and l. However, notice that the two definitions of anteriors are the same for AGs as there is arrowhead pointing to lines in AGs in both definitions, i.e. AGs are

anterior graphs. In addition, the two definitions for AGs are the same since if $i \in \text{ant}(\text{pa}(i) \cup \text{sp}(i)) \setminus \text{an}(\text{pa}(i) \cup \text{sp}(i))$ then condition 2 does not hold since by Lemma 2.2 there is an arrowhead pointing to a line in graph.

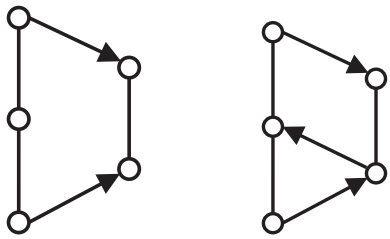
Acyclic directed mixed graphs. An *acyclic directed mixed graph* (ADMG) is an LMG that consists of arcs and arrows only and contains no direction-preserving cycles. ADMGs can be considered AAGs with no lines. In addition, an ADMG that does not contain an arc is a DAG. For a statistical discussion on a subclass of this graph see Spirtes et al. [1997].

Chain graphs and their independence interpretations. A graph $G = (V, E)$ with two types of edges (arrows and one symmetric type of edge, which can be lines or arcs) is a *chain graph* if it does not contain any cycle that contains at least one arrow and may contain symmetric edges and whose arrows are all pointing to one direction. It is implied from the definition that such graphs are characterised by having a node set that can be partitioned into numbered subsets forming so-called *chains*, which are connected subgraphs result by removing all arrows in the graph. All edges between nodes in the same chain are of the same type and symmetric, and all edges between different chains are arrows pointing from a chain with a higher number to one with a lower number. One can observe that if we replace every chain by a node, we obtain a DAG.

Regarding the graphical structure, chain graphs can be considered a special type of LMGs. For the purpose of defining independence model on graphs, however, separation criteria are needed. At least four different separation criteria for chain graphs, i.e. four different types of global Markov property have been discussed in the literature. Drton [2009] has classified them as the *LWF* or *block concentration* Markov property, the *AMP* or *concentration regression* Markov property, a Markov property that is dual to the AMP Markov property, and the *multivari-*

ate regression Markov property. It can be shown that among these the criterion that is used for multivariate regression Markov property is identical to the m-separation criterion when the chain components consist of arcs. Other separation criteria are not special cases of m-separation.

Undirected chain graphs. An *undirected chain graph* (UCG) is a chain graph with chain components connected by lines.



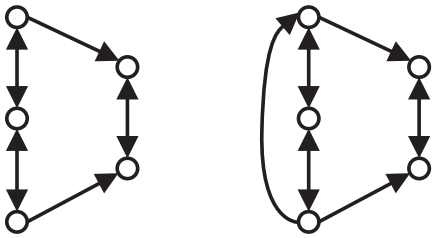
(a) (b)

Figure 2.10: (a) A UCG. (b) An LMG that is not undirected chain.

One can think of a UCG as a subclass of LMGs that uses the m-separation criterion. This associated global Markov property for a UCG is different from and of less interest than the four well-known global Markov properties discussed in Drton [2009]. This type of graph has not been used in the literature. This can be due to the fact that UCGs are not ribbonless.

Bidirected chain graphs and multivariate regression chain graphs. A *bidirected chain graph* (BCG) is a chain graph $G = (V, E)$ with chain components connected by arcs. For example, see Figure 2.11. As it was mentioned this class of graphs coincides with the class of graphs in statistics defined by Cox and Wermuth [1993] called *multivariate regression chain graphs*; see Wermuth and Cox [2004].

Undirected graphs and concentration graphs. The simplest subclass of LMGs is the class of undirected graphs. This class has been widely used in areas



(a) (b)

Figure 2.11: (a) A BCG. (b) An LMG that is not bidirected chain.

of mathematics, statistics, and computer science including networks and machine learning. An *undirected graph* (UG) is a graph that only consists of lines.

Undirected graphs are identical to the independence undirected graphs that have been widely used in the literature of graphical models. In some texts these are called *concentration graphs* since they are derived from the concentration matrix of multidimensional normal random variables. For a statistical discussion on this type of graph, see Lauritzen [1996].

Since UGs consist of one type of edge, a simplified version of the m-separation criterion can be applied to them. In this case the m-separation criterion is simplified to the ordinary separation that has been defined in the graph theory literature, i.e. A is separated from B by C if there is no path between A and B such that all its inner nodes are outside C.

Bidirected graphs and covariance graphs. Another simple subclass of LMGs is the class of bidirected graphs. This class has become actively used in statistics. A *bidirected graph* (BG) is a graph that only consists of arcs.

Bidirected graphs are identical to a type of independence graph that have been widely used in statistics. In some texts these are called a *covariance graphs* since they are derived from the covariance matrix of multidimensional normal random variables. For a statistical discussion on this type of graph, see Cox and Wermuth

[1993] and Wermuth and Cox [1998].

Since BGs consist of one type of edge, a simplified version of the m-separation criterion can be applied to them. In this case A is separated from B by C if there is no path between A and B such that all its inner nodes are in C.

Directed acyclic graphs. The class of DAGs, which has been discussed in Chapter 1, is also a subclass of LMGs; for a discussion on this type of graph see Lauritzen [1996].

For DAGs, since the only types of collider and non-collider nodes are the collision and non-collision nodes, the m-separation criterion is simplified to the d-separation criterion introduced by Pearl [1988]. In a directed acyclic graph G a path is said to be *d-connecting given C* if along it every collision node is in C or has a descendant in C and every other node is outside C. Furthermore, two disjoint sets of nodes A and B are said to be *d-separated* by C if and only if given C there is no d-connecting path between A and B, i.e. between a node in A and a node in B. Then we write $A \perp_d B \,|\, C$ if A and B are d-separated by C in G. This criterion is equivalent to the criterion defined in Chapter 1 for DAGs, i.e. we have the following Proposition:

Proposition 2.4. *[Marchetti and Wermuth, 2009] Let $G = (V, E)$ be a directed acyclic graph and A, B, and C be disjoint subsets of V. It holds that $A \perp_d B \,|\, C$ if and only if $A \perp_w B \,|\, C$.*

Proof. See Proposition 5 of Marchetti and Wermuth [2009]. □

For example, from Figure 1.2(a) we conclude that $A = \{k\}$ and $B = \{p, r\}$ are not d-separated by $C = \{i, j\}$ since $\langle p, h, i, l, k\rangle$ is a d-connecting path given C, which is the same as the conclusion that was reached by use of the partial-ancestor graph in Figure 1.2(b).

The hierarchy of subclasses of LMGs. The following diagram illustrates the hierarchy regarding inclusions of subclasses of LMGs. For example, one can observe from the diagram that BGs are a subclass of AGs, since these are a subclass of BCGs, and BCGs are a subclass of AGs.

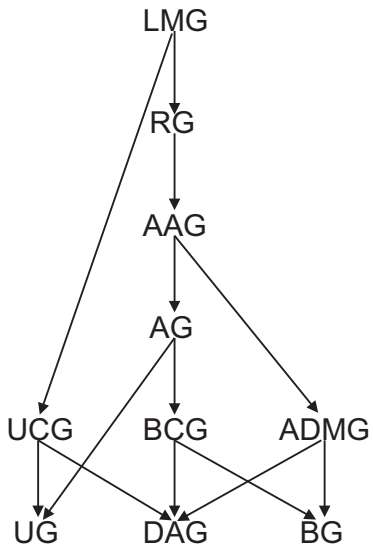


Figure 2.12: The hierarchy of subclasses of LMGs.

Proposition 2.5. *The hierarchy of subclasses of LMGs displayed in Figure 2.12 is valid.*

Proof. RGs and UCGs are subclasses of LMGs since they only have the three desired types of edge and are loopless. AAGs is a subclass of RGs since there is no arrowheads pointing to a line and no direction-preserving cycle in AAGs, hence ribbons are not permissible in AAGs. AGs is a subclass of AAGs since there are no arrowheads pointing to a line and no direction-preserving cycle in AGs. ADMGs is a subclass of AAGs since they do not contain lines and there is no direction-preserving cycle in them. BCGs is a subclass of AGs since they do not contain lines, there is no direction-preserving cycle in them, and an arc with one endpoint that is an ancestor of the other endpoint does not exist in BCGs.

UGs are trivially a subclass of AGs and UCGs. BGs are also trivially a subclass of BCGs and ADMGs. DAGs is a subclass of UCGs, BCGs, and ADMGs since they only contain arrows and there is no direction-preserving cycle in them. □

2.4 Maximality for ribbonless graphs

Definition of maximality. Here we start with simple definition of maximality. We show that there are other results by which maximality can be examined.

A graph G is called *maximal* with respect to separation criterion $\mathcal{S}$ if by adding any edge to G the independence model induced by $\mathcal{S}$ changes.

Since the independence models induced by separation criteria conform with the graphs (see condition 2 of the definition of separation criterion in Section 1.3.1) and adding an edge does not destroy any walks (see condition 1 of the same definition), for maximal graphs, adding an edge to the graph makes the independence model smaller. Therefore, in maximal graphs, every missing edge corresponds to at least one independence statement in the induced independence model in the sense that in $G = (V, E)$ if $i \not\sim j$ then $\langle i, j \,|\, C\rangle \in \mathcal{J}_{\mathcal{S}}(G)$ for some $C \subset V$.

In some texts, e.g., Wermuth [2011], a subclass containing maximal graphs is called *independence graphs*. Since henceforth we only deal with the subclasses of LMGs and the m-separation criterion, we may sometimes skip emphasising that maximality is with respect to $\perp_m$.

Verifying maximality of graphs. The direct translation of the definitions into formulation gives the lemma below.

Lemma 2.5. *A graph $G = (V, E)$ is maximal with respect to $\mathcal{S}$ if and only if, for every pair of non-adjacent nodes i and j of V, there exists a subset C of $V \setminus \{i, j\}$ such that $i \perp_{\mathcal{S}} j \,|\, C$.*

Proof. The result follows directly from the definition of maximality. □

We prove that among the subclasses of LMGs that have been defined in this chapter, RGs, AAGs, and AGs are not maximal with respect to $\perp_m$, while the rest are maximal.

Maximality of RGs. RGs are not maximal with respect to $\perp_m$. To observe this consider the RG in Figure 2.13 and Lemma 2.5. There is no C such that $i \perp_m j \,|\, C$. This is because if $k \in C$, the path $i \longrightarrow k \longleftrightarrow j$ is m-connecting given C, and if $k \notin C$, $i \longrightarrow k \longrightarrow j$ is m-connecting given C.

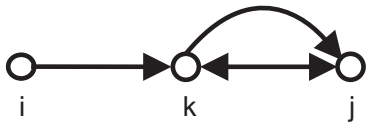


Figure 2.13: A non-maximal RG.

In Theorem 2.2 we give a condition under which an RG is maximal. We next show that from a non-maximal RG, how to generate a maximal RG with respect to $\perp_m$ that induces the same independence model as the RG, by adding edges to the RG.

A path $\langle j, q_1, q_2, \ldots, q_p, i\rangle$ is a *primitive inducing* path between i and j if and only for every n, $1 \leq n \leq p$,

i) q_n is a collider on the path; and

ii) $q_n \in \text{an}(\{i\} \cup \{j\})$.

For AGs, a primitive inducing path is an inducing path (defined in Section 3.4) with $M = C = \varnothing$. This is why the term primitive inducing path has been used. Figure 2.14 illustrates what a primitive inducing path looks like.

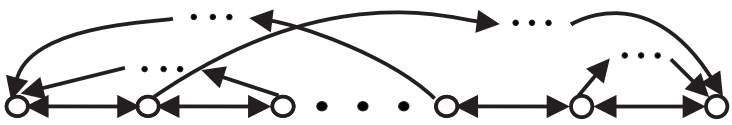


Figure 2.14: A primitive inducing path.

Proposition 2.6. *In a ribbonless graph, if there exists a primitive inducing path with no arrowheads pointing to its endpoints, then there is a line between the endpoints.*

Proof. For the primitive inducing path $\langle j, q_1, q_2, \ldots, q_p, i\rangle$, suppose that jq_1- and q_pi-edges are arrows pointing to q_1 and q_p. If $q_1 \in \mathrm{an}(j)$ then $\langle j, q_1, \ldots, j\rangle$ is a direction-preserving cycle. Similarly if $q_p \in \mathrm{an}(i)$ then $\langle i, q_p, \ldots, i\rangle$ is a direction-preserving cycle. If $q_1 \in \mathrm{an}(i)$ and $q_p \in \mathrm{an}(j)$ then $\langle j, q_1, \ldots, i, q_p, \ldots, j\rangle$ is a direction-preserving cycle. Since the graph is ribbonless there is an arrow from j to q_2 or an arrow from i to q_{p-1} in the graph, which generates a shorter primitive inducing path with no arrowheads at the endpoints on the path. Therefore, by induction, there is a line between i and j. □

Next we need the following lemmas. These also establish a pairwise Markov property for maximal RGs.

Lemma 2.6. *A non-collider node k on a path π between i and j in a ribbonless graph H is either in* $\mathrm{ant}(i) \cup \mathrm{ant}(j)$ *or an anterior of a collider node h on π. Moreover, the relevant subpath of π between k and i, j or h is an anterior path in H^*.*

Proof. For path $\pi = \langle i = i_0, i_1, \ldots, i_n = j\rangle$, there is at least no arrowheads pointing to a non-collider node $k = i_m$ from one side (say from i_{m-1}) on π. By moving towards i on the path as long as i_p, $1 \le p \le m-1$, is non-collider on the path, $k \in \mathrm{ant}(i_{p-1})$. This implies that if no i_p is collider then $k \in \mathrm{ant}(i)$, and hence the lemma follows. □

Lemma 2.7. *For nodes i and j in an RG that are not connected by any primitive inducing paths (hence $i \not\sim j$), it holds that $i \perp_m j \mid (\mathrm{ant}(i) \cup \mathrm{ant}(j)) \setminus \{i, j\}$.*

Proof. Suppose, for contradiction, that there is a shortest m-connecting path π between i and j given $(\mathrm{ant}(i) \cup \mathrm{ant}(j)) \setminus \{i, j\}$. If there is a non-collider node k on π then, by Lemma 2.6, k is either in $\mathrm{ant}(i) \cup \mathrm{ant}(j)$ through the nodes on π or it is an anterior of a collider node on π. But since π is m-connecting given $(\mathrm{ant}(i) \cup \mathrm{ant}(j)) \setminus \{i, j\}$, collider nodes are in $\mathrm{ant}(i) \cup \mathrm{ant}(j)$ themselves. Hence $k \in \mathrm{ant}(i) \cup \mathrm{ant}(j)$, which contradicts the fact that π is m-connecting. Therefore, all inner nodes of π should be colliders.

Now we know that all inner nodes of π are in $\mathrm{ant}(i) \cup \mathrm{ant}(j)$ and $i \not\sim j$. If, for a collider V-configuration $\langle r, l, s \rangle$ on π, $l \in (\mathrm{ant}(i) \cup \mathrm{ant}(j)) \setminus (\mathrm{an}(i) \cup \mathrm{an}(j))$ then, by Lemma 2.2 and since the graph is ribbonless, there is an endpoint-identical rs-edge to the V-configuration, which contradicts π being shortest. Therefore, $l \in \mathrm{an}(i) \cup \mathrm{an}(j)$, which implies that π is primitive inducing, again a contradiction. Therefore, there is no m-connecting path between i and j given $(\mathrm{ant}(i) \cup \mathrm{ant}(j)) \setminus \{i, j\}$, and hence $i \perp_m j \mid (\mathrm{ant}(i) \cup \mathrm{ant}(j)) \setminus \{i, j\}$. □

Now we give a necessary and sufficient condition for an RG to be maximal. The analogous result for ancestral graphs was proved in Richardson and Spirtes [2002].

Theorem 2.2. *A ribbonless graph H is maximal if and only if H does not contain any primitive inducing paths between non-adjacent pairs of nodes.*

Proof. Let $\pi = \langle i = i_0, i_1, \ldots, i_n = j \rangle$ be a primitive inducing path between i and j in H, and let C be a subset $V \setminus \{i, j\}$, where V is the node set of H. We need to show that there is an m-connecting path between i and j given C.

This is immediate if each internal node, i.e. each of $i_1, \ldots, i_{n-1}$, is in $C \cup \mathrm{an}(C)$ by just using π, so assume that this is not the case. Thus there is an internal

node of π not in $C \cup \mathrm{an}(C)$, and we may assume that there is one in $\mathrm{an}(i)$. Pick such a node i_q, $1 \leq q < n$, as far along the path to j as possible. Consider a direction-preserving path from i_q to i, and let P_1 denote the reverse of this path. Note that no internal node in P_1 is in $C \cup \mathrm{an}(C)$. Let π_1 be the part of π from i_q to j. If each internal node in this path is in $C \cup \mathrm{an}(C)$ then we are done by taking the path P_1 followed by π_1 (note that no node can be repeated since each internal node in π_1 is in $C \cup \mathrm{an}(C)$ and each internal node in P_1 is outside $C \cup \mathrm{an}(C)$). So suppose not. Let i_p be the first node in π_1 that is not in $C \cup \mathrm{an}(C)$. Then $i_p \notin \mathrm{an}(i)$ (by the way i_q was chosen), so $i_p \in \mathrm{an}(j)$. Let π_2 be the part of π from i_q to i_p, and let P_2 be a direction-preserving path from i_p to j. Note that no internal node in P_2 is in $C \cup \mathrm{an}(C)$. If P_1 and P_2 have no intersection then much as above we obtain an m-connecting path given C by taking P_1 followed by π_2, followed by P_2. If P_1 and P_2 do intersect then we obtain an m-connecting path as required by following P_1 up to the first node on P_2 and then following P_2.

By letting $C = (\mathrm{ant}(i) \cup \mathrm{ant}(j)) \setminus \{i, j\}$ for every non-adjacent nodes i and j, the other direction follows from Lemmas 2.5 and 2.7. □

The following algorithm generates a maximal RG from an RG:

Algorithm 2.2. *(Generating a maximal RG from a ribbonless graph H)*
Start from H.

Generate an endpoint-identical edge between i and j to a primitive inducing path between non-adjacent i and j, i.e. (by using Proposition 2.6)

1. *Generate an arrow from j to i for primitive inducing paths between non-adjacent i and j where there is no arrowhead pointing to j on the path.*
2. *Generate an arc between i and j for primitive inducing paths between non-adjacent i and j where there are arrowheads pointing to i and j on the path.*

Apply the previous steps until no other edge can be generated.

Lemma 2.8. *The order of applying the steps of Algorithm 2.2 does not affect the final graph obtained.*

Proof. The result follows from the fact that adding an edge between endpoints of a primitive inducing path does not destroy other primitive inducing paths in the graph as well as the primitive inducing paths that might be generated by the algorithm in the next iterations. Hence at each iteration of the algorithm if there are two non-adjacent pairs of nodes connected by primitive inducing paths (even the newly generated ones) then it does not matter which one to be chosen first. □

We prove that the graph generated by Algorithm 2.2 is maximal.

Proposition 2.7. *From a given ribbonless graph H, Algorithm 2.2 generates a maximal RG that induces the same independence model as that of H.*

Proof. The proof has three parts. In part I we prove that the generated graph is ribbonless. In part II we prove that the graph is maximal. In part III we prove that the generated graph induces the same independence model as that of H.

Part I. The generated graph is ribbonless: Consider one fixed order of applying the steps of Algorithm 2.2. Consider also the collection $\mathcal{C}$ of added edges by this algorithm. For any iteration G of the algorithm we say that a ribbon $\langle i, k, j\rangle$ is *durable* if the corresponding endpoint-identical ij-edge is not in $\mathcal{C}$. Now, we show by contradiction that no durable ribbon will be created.

Consider the first iteration of the algorithm where such a ribbon is created. Denote this by $\langle j, i, i_1\rangle$. We need to consider the two following cases:

Case I.1. Suppose that zw is the added edge at this iteration, and is an arrow from z to w, which is not on a ribbon but is on a direction-preserving path or cycle $\pi' = \langle i, \ldots, z, w, \ldots, l\rangle$ that generates the ribbon $\langle j, i, i_1\rangle$. (l is an endpoint of a line or $l = i$.) Suppose also that the zw-edge is generated by the primitive

inducing path $\pi = \langle z = g_0, g_1, g_2, \ldots, g_s, g_{s+1} = w \rangle$. Notice that the zg_1-edge is an arrow from z to g_1. In this case, if $g_1 \in \text{an}(z)$ then z and g_1 are on a direction-preserving cycle and if $g_1 \in \text{an}(w)$ then i is an ancestor of a node that is an endpoint of a line or on a direction-preserving path. Therefore, before adding the zw-edge there is a ribbon $\langle j, i, i_1 \rangle$ in graph, and the zw-edge has not generated a new ribbon. Therefore, this case is not possible.

Case I.2. Suppose that the added edge is the ij-edge, generated by a primitive inducing path $\pi = \langle j = q_0, q_1, q_2, \ldots, q_p, q_{p+1} = i \rangle$ in the preceding iteration of the algorithm, to generate the ribbon $\langle j, i, i_1 \rangle$. We observe that $\langle q_p, i, i_1 \rangle$ is also a ribbon. Thus, by the assumption, this ribbon is not durable, and the corresponding end-point identical $q_p i_1$-edge must be added during the algorithm. In addition, for all $q_n \in \text{an}(i)$, the ribbon $\langle q_{n-1}, q_n, q_{n+1} \rangle$ is not durable and the corresponding endpoint-identical $q_{n-1}q_{n+1}$-edge must be added during the algorithm. Denote the $q_n \in (\text{an}(j) \setminus \text{an}(i))$ by $r_1, \ldots, r_m$.

Now $\langle j, r_1, r_2, \ldots, r_m, i_1 \rangle$ is a primitive inducing path at some iteration of the algorithm, and hence an endpoint-identical ji_1-edge will be added by a later iteration of the algorithm. This is also endpoint-identical to $\langle j, i, i_1 \rangle$. This contradicts the assumption that $\langle j, i, i_1 \rangle$ is durable.

Part II. Maximality follows from Theorem 2.2 since there is no primitive inducing path in the generated graph.

Part III. Now we prove that the generated maximal RG induces the same independence model as the one for H. We use intermediate graphs to generate the final graph from H that have each been generated by adding one edge to the previous graph by one of the steps of Algorithm 2.2. We denote these graphs by $\langle H = H_0, H_1, \ldots, H_n \rangle$. Since Algorithm 2.2 only adds edges, it is enough to prove that no independence statement has disappeared by adding an ij-edge to an arbitrary intermediate graph H_s generated by the algorithm. We know that there is a

primitive inducing path $\pi = \langle j = q_0, q_1, q_2, \ldots, q_p, q_{p+1} = i \rangle$ between i and j in H_s that is endpoint-identical to the ij-edge. We need to consider two cases here. In case III.1 we consider the case that the generated ij-edge is on an m-connecting path in H_{s+1} and find an m-connecting path in H_s. In case III.2 we consider the case that the generated ij-arrow is on a direction-preserving path from a collider node on an m-connecting path in H_{s+1} and find a direction-preserving path from and to the same nodes in H_s.

Case III.1. Suppose that the ij-edge is on an m-connecting path given C in H_{s+1} and π is a shortest among the endpoint-identical primitive inducing paths between i and j. Notice that on this path there is at least an arrowhead pointing to i or j. If there are arrowheads at both endpoints then the same method provided in the proof of Theorem 2.2 gives an endpoint-identical m-connecting path between i and j.

Thus suppose that there is an arrowhead at i but not at j on the path. In this case we know that $q_1 \in \text{an}(i)$. We trace an endpoint-identical m-connecting path between i and j by the following method: We start from j, and proceed to q_1 through the jq_1-edge. If no node on the direction-preserving path from q_1 to i is in C then we move through the direction-preserving path to i and we are done. Thus suppose that there is a node in C and, therefore, $q_1 \in \text{an}(C)$. This implies also that $j \in \text{an}(C)$ and, therefore for each $q_t \in \text{an}(i)$, $1 \leq t \leq p$, it holds that $q_t \in \text{an}(C)$. Now from q_1 we proceed to q_2. Again if there is a direction-preserving path from q_2 to i without any node in C we move to i through this path, otherwise, $q_2 \in \text{an}(C)$ and we move to q_3. By induction, we obtain an endpoint-identical m-connecting path given C to the ij-edge in H_{s+1}.

Case III.2. Suppose that the generated arrow from j to i in H_{s+1} is on a direction-preserving path from a collider node (on an m-connecting path) k to a node l. We know that there is a shortest primitive inducing path $\langle j, q_1, q_2, \ldots, q_p, i \rangle$ in

H_s such that the jq_1-edge is an arrow from j to q_1 and the $q_p i$-edge is an arc. Notice that $q_1 \in \text{an}(i)$ since by part I.2 of this proof we know that H_s does not contain ribbons of type (ii), and therefore, if $q_1 \in \text{an}(j)$ then $\langle j, q_1, q_2 \rangle$ is a ribbon of type (ii) unless there is an endpoint-identical jq_2-edge, which makes a shorter primitive inducing path. Therefore, instead of the generated ij-arrow, in H_s one can use the direction-preserving path that consists of the jq_1-arrow and the direction-preserving path from q_1 to i. □

Computational complexity of the algorithm. To discuss computational complexity in this thesis, we consider the input graph to be a vector of nodes and a vector of edges, which are pairs of nodes.

Notice that the given algorithm can be run in polynomial time. For example, for each missing edge ij we look for ancestors of i and j. This can be done in $|\text{an}(i) \cup \text{an}(j)|^2 e$ steps, where e is the number of edges of the graph. It is enough then to deduce whether there is a collision path between i and j in the subgraph induced by $\text{an}(i) \cup \text{an}(j)$. This is the same as the known problem of discerning whether there exists an m-connecting path between two nodes. This can be achieved within $O(|\text{an}(i) \cup \text{an}(j)|^3)$. Therefore, in the worst case scenario there are $\binom{n}{2}(n^2 e + n^3)$ steps and the complexity (assuming that the graphs are connected) is $O(n^5)$, where n is the number of nodes of the graph.

Maximality of AAGs and ADMGs. AAGs, like RGs, are not maximal with respect to $\perp_m$. To observe this one can consider the AAG in Figure 2.13. Since primitive inducing paths are permissible in AAGs, all the maximality results discussed in this section for RGs are valid for AAGs.

Proposition 2.8. *From a given acyclic armorless graph H, Algorithm 2.2 generates a maximal AAG that induces the same independence model as H.*

Proof. Using Proposition 2.7, it is enough to prove that the generated graph is

an AAG. We use intermediate graphs to generate the final graph from H that have each been generated by adding one edge to the previous graph by one of the steps of Algorithm 2.2. We denote these graphs by $\langle H = H_0, H_1, \ldots, H_n \rangle$.

We prove that the generated graph has no arrowheads pointing to lines. We prove this by induction along the sequence. For H the result holds by definition. For the inductive step, suppose, for contradiction, that H_m is a graph with an arrowhead pointing to line. The algorithm does not generate any lines, hence suppose that there is a generated ij-edge with arrowhead pointing to j, which is an endpoint of a line in H_m. This implies that there is an arrowhead pointing to j on another edge (on the primitive inducing path) in H_{m-1}, a contradiction by the induction hypothesis.

Now we prove that there is no direction-preserving cycle in the generated graph. We prove this by induction along the sequence. For H the result holds by definition. For the inductive step, suppose, for contradiction, that H_r has a direction-preserving cycle. The generated edge is an arrow, which we suppose is from j to i. Therefore, in H_{r-1} there is a shortest primitive inducing path $\pi = \langle j = q_0, q_1, q_2, \ldots, q_p, q_{p+1} = i \rangle$ with no arrowheads at j on π. This implies that $q_1 \in \mathrm{an}(i)$ since if $q_1 \in \mathrm{an}(j)$ then $\langle j, q_1, q_2 \rangle$ is a ribbon unless there is an arrow from j to q_2, which contradicts π being shortest. Now in H_{r-1} by using the jq_1-arrow and the direction-preserving path from q_1 to i (instead of the ij-edge) one can find a direction-preserving cycle in H_{r-1}, a contradiction with the induction hypothesis. □

Corollary 2.2. *From a given acyclic directed mixed graph H, Algorithm 2.2 generates a maximal ADMG that induces the same independence model as H.*

Proof. The result follows from the fact that Algorithm 2.2 does not generate any lines. □

Maximality of AGs. AGs are not maximal with respect to $\perp_m$. For example, the AG in Figure 2.15, which contains the primitive inducing path $\langle i, h, k, j \rangle$.

Figure 2.15: A non-maximal AG.

Some primitive inducing paths, such as the primitive inducing path in Figure 2.13, are, however, not permissible in AGs. We call the primitive inducing paths that are permissible in AGs *ancestral primitive inducing paths*, i.e. $\pi = \langle i_0, i_1, \ldots, i_{n-1}, i_n \rangle$ is called an ancestral primitive inducing path if it consists of only arcs, $i_1 \in \text{an}(i_n)$, $i_{n-1} \in \text{an}(i_0)$, and for $k \in \{2, \ldots, n-2\}$, either $i_k \in \text{an}(i_0)$ or $i_k \in \text{an}(i_n)$. Figure 2.15 is the simplest example of an ancestral primitive inducing path.

Now conditions for maximality of AGs can be simplified. We then show that from the non-maximal AG how to generate a maximal AG with respect to $\perp_m$ that induces the same independence model as the AG by adding edges to the AG. These results conform with and have been proven in Richardson and Spirtes [2002].

Proposition 2.9. *[Richardson and Spirtes, 2002] An ancestral graph H is maximal with respect to $\perp_m$ if and only if there are not non-adjacent nodes in H connected by an ancestral primitive inducing path.*

The following algorithm generates a maximal AG from an AG:

Algorithm 2.3. *(Generating a maximal AG from an ancestral graph H)*
Start from H.

Generate an arc between non-adjacent nodes i and j connected by ancestral primitive inducing paths.

Apply the previous step until no other edge can be generated.

Proposition 2.10. *[Richardson and Spirtes, 2002] From a given ancestral graph H, Algorithm 2.3 generates a maximal AG that induces the same independence model as H.*

Maximality of BCGs, UCGs, UGs, BGs, and DAGs. We show that BCGs, UCGs, UGs, BGs, and DAGs are maximal with respect to $\perp_m$.

Proposition 2.11. *BCGs, UCGs, UGs, BGs, and DAGs are maximal with respect to $\perp_m$.*

Proof. Since primitive inducing paths are not permissible in these graphs, the result follows from Theorem 2.2. □

For BCGs, UGs, BGs, and DAGs the following establish simplified pairwise Markov properties.

Proposition 2.12. *Let i and j be nodes in a BCG. If $i \not\sim j$ then $i \perp_m j \,|\, \mathrm{an}(i) \cup \mathrm{an}(j) \setminus \{i, j\}$.*

Proof. The result follows from Lemma 2.7 and the fact that BCGs do not contain lines and hence $\mathrm{ant}(i) = \mathrm{an}(i)$. □

Proposition 2.13. *[Lauritzen, 1996] Let i and j be nodes in an undirected graph $H = (N, F)$. If $i \not\sim j$ then $i \perp_m j \,|\, N \setminus \{i, j\}$.*

Proof. The result follows from Lemma 2.7 and the fact that UGs only contain lines and hence $\mathrm{ant}(i) = N \setminus \{i\}$. □

Proposition 2.14. *[Kauermann, 1996] Let i and j be nodes in a BG. If $i \not\sim j$ then $i \perp_m j$.*

Proof. The result follows from Lemma 2.7 and the fact that BGs do not contain lines or arrows and hence $\mathrm{ant}(i) = \varnothing$. □

Proposition 2.15. *Let i and j be nodes in a DAG. If $i \not\sim j$ then $i \perp_m j \,|\, \mathrm{an}(i) \cup \mathrm{an}(j) \setminus \{i, j\}$.*

Proof. The result follows from Lemma 2.7 and the fact that DAGs do not contain lines and hence $\mathrm{ant}(i) = \mathrm{an}(i)$. □

Maximality of anterior graphs. We also have the following lemma for the anterior graphs.

Lemma 2.9. *Let H be a ribbonless graph and H^* its anterior graph. Then if H is maximal, so is H^*.*

Proof. If, for contradiction, H^* is maximal, then Theorem 2.2 implies that there is a primitive inducing path in H^* between non-adjacent nodes i and j. Consider a shortest of such primitive inducing paths between i and j and denote it by π. We know that all inner nodes of π are colliders in H^*. This trivially implies that all inner nodes of π are colliders in H too. In addition, each inner node k on π is in $\mathrm{an}(\{i, j\})$ in H^*. In H, $k \in \mathrm{an}(\{i, j\})$ unless an arrow on the direction-preserving path from k to i or j is an arc turning into an arrow in H^*. In this case k is an ancestor of a node that is an endpoint of a line. Hence the V-configuration $\langle h, k, l \rangle$ on π is a ribbon unless there is an endpoint-identical hl-edge to the V-configuration, which contradicts the fact that π is a shortest. Therefore, π is a primitive inducing path in H, a contradiction. Hence H^* is maximal. □

2.5 Markov properties for ribbonless graphs

Goal of the section. A global Markov property that uses the m-separation criterion and a pairwise Markov property were defined in Richardson and Spirtes [2002] for maximal ancestral graphs without considering the conditions under which their equivalence holds. We use a generalisation of these Markov properties for maximal RGs, which contains maximal ancestral graphs as a subclass,

and prove their equivalence for compositional graphoids as a main result of this section. This result is a direct generalisation of the similar result of [Pearl, 1988] for undirected graphs and graphoids. It has been mentioned as a conjecture in Kang and Tian [2009], who took the composition property into account and discussed several Markov properties. The non-trivial direction of this result states that starting with pairwise Markov property statements, one can generate all independence statements implied by the global Markov property from the compositional graphoid axioms.

Global and pairwise Markov property for RGs. Consider an independence model $\mathcal{J}^H$ defined over the node set of a ribbonless graph H. Using the m-separation criterion, a global Markov property has been defined on RGs and its subclasses. For disjoint subsets A, B, and C of the node set of ribbonless graph H, the independence model $\mathcal{J}^H$ satisfies the global Markov property if $A \perp_m B \,|\, C$ implies $\langle A, B \,|\, C\rangle \in \mathcal{J}^H$.

In the last section we also established a pairwise Markov property on maximal RGs and its subclasses. For nodes i and j, the independence model $\mathcal{J}^H$ satisfies the pairwise Markov property if $i \not\sim j$ implies $\langle i, j \,|\, (\mathrm{ant}(i) \cup \mathrm{ant}(j)) \setminus \{i, j\}\rangle \in \mathcal{J}^H$.

For example, for the graph in Figure 2.16, the pairwise Markov property would imply that $\langle i, m \,|\, \{k, l, h\}\rangle$ as $\mathrm{ant}(i) = \{k, l, h, m\}$ and $\mathrm{ant}(m) = \{l, h\}$. It would also imply that $\langle l, p \,|\, \{h, m\}\rangle$.

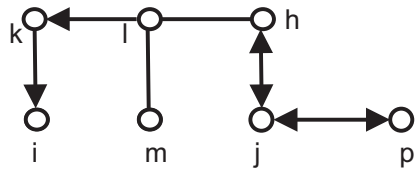


Figure 2.16: A loopless mixed graph H. The pairwise Markov property for H implies, e.g., $\langle i, m \,|\, \{k, l, h\}\rangle$. The global Markov property, e.g., implies $\langle \{i, k\}, j \,|\, l\rangle$.

Clearly, the independence model $\mathcal{J}_m(H)$, induced by m-separation, always satisfies the global Markov property. We also have the following proposition:

Proposition 2.16. *Let H be a loopless mixed graph. The independence model $\mathcal{J}_m(H)$ then satisfies the pairwise Markov property if and only if H is maximal.*

Proof. The result follows from Lemma 2.5, Lemma 2.7, and Theorem 2.2. □

Equivalence of pairwise and global Markov properties. Before introducing the main result of this section, we introduce two lemmas.

Lemma 2.10. *Let $\mathcal{J}$ be a compositional graphoid over a set V and M and C be disjoint subsets of V. It then holds that the marginal independence model $\alpha(\mathcal{J}, M) = \{\langle A, B \,|\, C\rangle : \langle A, B \,|\, C\rangle \in \mathcal{J} \text{ and } (A \cup B \cup C) \cap M = \varnothing\}$ is a compositional graphoid.*

Proof. All the six compositional graphoid properties for $\alpha(\mathcal{J}, M)$ follows trivially from the facts that for A, B, and C such that $(A \cup B \cup C) \cap M = \varnothing$, $\langle A, B \,|\, C\rangle \in \alpha(\mathcal{J}, M)$ if and only if $\langle A, B \,|\, C\rangle \in \mathcal{J}$, and $\mathcal{J}$ satisfies the six properties. □

The following lemma deals with the combination of two m-connecting paths in anterior graphs.

Lemma 2.11. *Let H^* be the anterior graph of a ribbonless graph H and suppose that there are paths $\pi_1 = \langle i = i_0, i_1, \ldots, i_n, h\rangle$ between i and h and $\pi_2 = \langle h, j_m, j_{m-1}, \ldots, j_0 = j\rangle$ between h and j which are m-connecting given C. The combination $\pi_{12} = \pi_1 \circ \pi_2$ is then an m-connecting path between i and j given C in each of the following mutually exclusive situations:*

a1) *$\langle i_n, h, j_m\rangle$ is a collider and $h \in C \cup \mathrm{an}(C)$;*

a2) *$i_n = j_m$ with an arrowhead pointing to h on the $i_n h$-edge and $h \in C \cup \mathrm{an}(C)$;*

b1) *$\langle i_n, h, j_m\rangle$ is a non-collider and $h \notin C$;*

b2) *$i_n = j_m$ with no arrowhead pointing to h on the $i_n h$-edge.*

Proof. Let $\pi_{12} = \pi_1 \circ \pi_2 = \langle i, \ldots, k, j_{q-1}, \ldots, j \rangle$ be the combination of π_1 and π_2. If $k = h$ and either a1) or b1) holds then the conclusion is obvious. The cases a2) or b2) are only relevant when $k \neq h$.

Next consider the situation where $k \neq h$. Since π_1 and π_2 are m-connecting, for π_{12} to be m-connecting we only need to check the V-configuration $\langle i_{p-1}, k, j_{q-1} \rangle$. We have to deal with two cases:

Case 1: $\langle i_{p-1}, k, j_{q-1} \rangle$ is a non-collider. In this case there is no arrowhead pointing to k from at least one of i_{p-1} or j_{q-1}. This means that $\langle i_{p-1}, k, i_{p+1} \rangle$ on π_1 or $\langle j_{q-1}, k, j_{q+1} \rangle$ on π_2 is a non-collider, and since π_1 and π_2 were both m-connecting we have $k \notin C$. Hence π_{12} is m-connecting.

Case 2: $\langle i_{p-1}, k, j_{q-1} \rangle$ is a collider. We need to consider the following two subcases:

Case 2.1. If $\langle i_{p-1}, k, j_{q-1} \rangle$ is a collider and any of $\langle i_{p-1}, k, i_{p+1} \rangle$ or $\langle j_{q-1}, k, j_{q+1} \rangle$ is also a collider then $k \in C \cup \text{an}(C)$ and π_{12} is m-connecting.

Case 2.2. If $\langle i_{p-1}, k, j_{q-1} \rangle$ is a collider but $\langle i_{p-1}, k, i_{p+1} \rangle$ and $\langle j_{q-1}, k, j_{q+1} \rangle$ are both non-colliders then by Lemma 2.6, the subpath of π_1 between k and a collider node l_1 or h is an anterior path and similarly for π_2, l_2, and h. However, since H^* is an anterior graph and there are arrowheads pointing to k, these anterior paths must be direction-preserving and thus $k \in \text{an}(l_1) \cup \text{an}(h)$ and $k \in \text{an}(l_2) \cup \text{an}(h)$. Now we have the two following further cases:

Case 2.2.1: One of the subpaths of π_1, π_2 from k to l_1, l_2 is direction-preserving. Because π_1 and π_2 are m-connecting we must have l_1 or l_2 in $C \cup \text{an}(C)$. Thus $k \in \text{an}(C)$ and π_{12} is m-connecting.

Case 2.2.2: Both subpaths of π_1 and π_2 from k to h are direction-preserving.

Then $\langle i_n, h, j_m \rangle$ is collider or $i_n = j_m$ with an arrowhead pointing to h on the $i_n h$-edge. Thus if a1) or a2) holds π_{12} is m-connecting since then $h \in C \cup \mathrm{an}(C)$. In addition, if b1) or b2) holds this case is not possible. □

We are now ready to establish the main result of this chapter.

Theorem 2.3. *Let H be a maximal ribbonless graph. If an independence model $\mathcal{J}^H$ over the node set of H is a compositional graphoid, then $\mathcal{J}^H$ satisfies the pairwise Markov property if and only if it satisfies the global Markov property.*

Proof. ($\Leftarrow$) If $\mathcal{J}^H$ is a compositional graphoid and satisfies the global Markov property it follows from Theorem 2.2 and Lemma 2.7 that it satisfies the pairwise Markov property.

($\Rightarrow$) Now suppose that $\mathcal{J}^H$ satisfies the pairwise Markov property and compositional graphoid axioms. For subsets A, B, and C of the node set of H, we should prove that $A \perp_m B \,|\, C$ implies $\langle A, B \,|\, C \rangle \in \mathcal{J}^H$. By composition, it is sufficient to show this when A and B are singletons, i.e. that $i \perp_m j \,|\, C$ implies $\langle i, j \,|\, C \rangle \in \mathcal{J}^H$.

Further we observe that it is sufficient to establish the result in the case when $H = H^*$ is itself an anterior graph. Proposition 2.1 gives that $A \perp_m B \,|\, C$ in H implies $A \perp_m B \,|\, C$ in H^*. In addition, by Lemma 2.9, H^* is a maximal graph. Moreover, H and H^* have the same anterior sets, and therefore the same pairwise Markov property. Hence in the following we assume that $H = H^*$ is an anterior graph.

We prove the result in two main parts. In part I we prove the result in the case that $C \subseteq \mathrm{ant}(i) \cup \mathrm{ant}(j)$. In part II we use the result of part I to establish the general case.

Part I. Suppose that $C \subseteq \mathrm{ant}(i) \cup \mathrm{ant}(j)$. We use induction on the number of nodes of the graph. The induction base for a graph with two nodes is trivial. Thus suppose that the result holds for all anterior graphs with fewer than n nodes

and assume that H^* has n nodes.

Let $D = \{i\} \cup \{j\} \cup \text{ant}(i) \cup \text{ant}(j)$ and $M = V \setminus D$, where V is the node set of the graph. First in case I.1 we suppose that $M \neq \varnothing$, and then in case I.2 we suppose that $M = \varnothing$.

Case I.1. Consider $H^*[D]$ to be the subgraph induced by D. Consider the marginal independence model $\alpha(\mathcal{J}^H, M) = \{\langle A, B \,|\, C\rangle : \langle A, B \,|\, C\rangle \in \mathcal{J}^H \text{ and } (A \cup B \cup C) \cap M = \varnothing\}$ defined over D. By Lemma 2.10, $\alpha(\mathcal{J}^H, M)$ is a compositional graphoid. In addition, it satisfies the pairwise Markov property: This is because two non-adjacent nodes l_1 and l_2 in $H^*[D]$ are non-adjacent in H^* and by the pairwise Markov property for $\mathcal{J}^H$, $\langle l_1, l_2 \,|\, \text{ant}_{H^*}(l_1) \cup \text{ant}_{H^*}(l_2) \setminus \{l_1, l_2\}\rangle \in \mathcal{J}^H$, where ant_{H^*} is the anterior set in H^*. We know that $\text{ant}_{H^*}(l_1) \cup \text{ant}_{H^*}(l_2) \subseteq D$ and hence $\text{ant}_{H^*}(l_1) \cup \text{ant}_{H^*}(l_2) \cap M = \varnothing$. In addition, for a node l in $H^*[D]$, $\text{ant}_{H^*}(l) = \text{ant}_{H^*[D]}(l)$. Therefore, $\langle l_1, l_2 \,|\, \text{ant}_{H^*[D]}(l_1) \cup \text{ant}_{H^*[D]}(l_2) \setminus \{l_1, l_2\}\rangle \in \alpha(\mathcal{J}^H, M)$.

We also know that $i \perp_m j \,|\, C$ in H^* implies $i \perp_m j \,|\, C$ in $H^*[D]$ since there is no m-connecting path between i and j given C in H^* and by removing nodes and edges from H^* no new m-connecting paths are generated. Therefore, by the induction hypothesis $\langle i, j \,|\, C\rangle \in \alpha(\mathcal{J}^H, M)$. This implies that $\langle i, j \,|\, C\rangle \in \mathcal{J}^H$.

Case I.2. Now suppose that $M = \varnothing$ and thus the node set of H^* is $D = \{i\} \cup \{j\} \cup \text{ant}(i) \cup \text{ant}(j)$. We prove the result by reverse induction on $|C|$: For the base, $C = V \setminus \{i, j\} = \text{ant}(i) \cup \text{ant}(j) \setminus \{i, j\}$ and the result follows trivially from the pairwise Markov property.

For the inductive step, consider a node $h \notin C$. We want to show that h is not m-connected to both i and j: Suppose, for contradiction, there are m-connecting paths $\pi_1 = \langle i, i_1, \ldots, i_n, h\rangle$ and $\pi_2 = \langle h, j_m, j_{m-1}, \ldots, j_0 = j\rangle$ given C. If b1) or b2) of Lemma 2.11 hold then i and j are m-connected given C which contra-

dicts $i \perp_m j \,|\, C$. So we need only consider the cases where $\langle i_n, h, j_m \rangle$ is collider or $i_n = j_m$ with an arrowhead pointing to h on the $i_n h$-edge. However, we know that $h \in \text{ant}(i)$ or $h \in \text{ant}(j)$. Because of symmetry between i and j suppose that $h \in \text{ant}(i)$. Since H^* is an anterior graph and there is an arrowhead pointing to h we have $h \in \text{an}(i)$. Hence there is a direction-preserving path π from h to i. If no node on π is in C then b1) or b2) of Lemma 2.11 implies that the combination of π and π_2 is an m-connecting path between i and j, again a contradiction. If there is a node on π that is in C then $h \in \text{an}(C)$ and again, by a1) and a2) of Lemma 2.11, i and j are m-connected given C, again a contradiction.

We conclude that, given C, h is not m-connected to both i and j. By symmetry suppose that $i \perp_m h \,|\, C$.

We also have that $i \perp_m j \,|\, C$. Since $\mathcal{J}_m(H^*)$ is a compositional graphoid (Theorem 2.1) the composition property gives that $i \perp_m \{j, h\} \,|\, C$. By weak union for $\perp_m$ we obtain $i \perp_m j \,|\, \{h\} \cup C$ and $i \perp_m h \,|\, \{j\} \cup C$. By the induction hypothesis we obtain $\langle i, j \,|\, \{h\} \cup C \rangle \in \mathcal{J}^H$ and $\langle i, h \,|\, \{j\} \cup C \rangle \in \mathcal{J}^H$. By intersection we get $\langle i, \{j, h\} \,|\, C \rangle \in \mathcal{J}^H$. By decomposition we finally obtain $\langle i, j \,|\, C \rangle \in \mathcal{J}^H$.

Part II. We now prove the result in the general case by induction on $|C|$. The base, i.e. the case that $|C| = 0$, follows from part I. To prove the inductive step we can assume that $C \nsubseteq \text{ant}(i) \cup \text{ant}(j)$, since otherwise part I implies the result.

We first show that if $C \nsubseteq \text{ant}(i) \cup \text{ant}(j)$ then there is a node l in C such that $i \perp_m j \,|\, C \setminus \{l\}$: Let first $l' \in C \setminus (\text{ant}(i) \cup \text{ant}(j))$ be arbitrary. If there is an $l'' \in C \setminus (\text{ant}(i) \cup \text{ant}(j))$ so that $l' \in \text{ant}(l'')$ and $l'' \notin \text{ant}(l')$ then replace l' by l'', and repeat this process until it terminates, the latter being ensured by transitivity of ant (Lemma 2.1) and the finiteness of C. Thus we eventually obtain an l so that if $l \in \text{ant}(\tilde{l})$ for $\tilde{l} \in C \setminus (\text{ant}(i) \cup \text{ant}(j))$ then we also have $\tilde{l} \in \text{ant}(l)$.

Suppose, for contradiction, that there is a shortest m-connecting path π between i and j given $C\setminus\{l\}$. If l is not on π or is a collider on π then π is also m-connecting given C. Therefore, l is a non-collider on π. This, together with $l \notin \mathrm{ant}(i)\cup\mathrm{ant}(j)$, by using Lemma 2.6, implies that l is an anterior of a collider node p on π. Since π is m-connecting, $p \in C \cup \mathrm{an}(C)$. Thus there is an $\tilde{l} \in C$ so that $p = \tilde{l}$ or $p \in \mathrm{an}(\tilde{l})$. Transitivity of anterior sets and the fact that $l \notin (\mathrm{ant}(i) \cup \mathrm{ant}(j))$ now imply that $\tilde{l} \in C \setminus (\mathrm{ant}(i)\cup\mathrm{ant}(j))$. The construction of l implies $\tilde{l} \in \mathrm{ant}(l)$ which again implies that $\tilde{l} \in \mathrm{an}(l)$ and $l \in \mathrm{an}(\tilde{l})$ and thus the collider V-configuration containing p is a ribbon of type (ii), unless its endpoints are adjacent with an endpoint-identical edge, which implies that π is not a shortest m-connecting path, a contradiction.

We now have that either $i \perp_m l \,|\, C \setminus \{l\}$ or $j \perp_m l \,|\, C \setminus \{l\}$ since otherwise, by Lemma 2.11 there is an m-connecting path between i and j given $C \setminus \{l\}$ in the case that l is a non-collider or given C in the case that l is a collider node. Because of symmetry suppose that $i \perp_m l \,|\, C \setminus \{l\}$. By the induction hypothesis we have $\langle i, j \,|\, C \setminus \{l\}\rangle \in \mathcal{J}^H$ and $\langle i, l \,|\, C \setminus \{l\}\rangle \in \mathcal{J}^H$. By the composition property we get $\langle i, \{j, l\} \,|\, C \setminus \{l\}\rangle \in \mathcal{J}^H$. The weak union property implies $\langle i, j \,|\, C\rangle \in \mathcal{J}^H$. □

Necessity of compositional graphoid axioms for the equivalence of pairwise and global Markov properties. Theorem 2.3 states that, for the equivalence of pairwise and global Markov properties, the six compositional graphoid axioms are sufficient. In fact, in general, for the mentioned equivalence, all six axioms are also necessary. The graphs in Figure 2.17 show that the intersection and composition properties are necessary for the equivalence of pairwise and global Markov properties.

For H_1, if $\mathcal{J}^{H_1}$ satisfies the pairwise Markov property then $\langle i, k \,|\, \{j, l\}\rangle$, $\langle i, l \,|\, \{j, k\}\rangle$, and $\langle k, l \,|\, \{i, j\}\rangle$ are in $\mathcal{J}^{H_1}$. One can observe that none of the semigraphoid axioms or the composition property can be used to imply $\langle i, \{k, l\} \,|\, j\rangle \in \mathcal{J}^{H_1}$. The

intersection property is the only axiom that implies the result.

For H_2, if $\mathcal{J}^{H_2}$ satisfies the pairwise Markov property then $\langle i, k \,|\, \varnothing\rangle$, $\langle i, l \,|\, \varnothing\rangle$, and $\langle k, l \,|\, \varnothing\rangle$ are in $\mathcal{J}^{H_2}$. One can observe that none of the semigraphoid axioms or the intersection property can be used to imply $\langle i, \{k, l\} \,|\, \varnothing\rangle \in \mathcal{J}^{H_2}$. The composition property is the only axiom that implies the result.

For H_3, if $\mathcal{J}^{H_3}$ satisfies the pairwise Markov property then $\langle i, k \,|\, \varnothing\rangle$, $\langle i, l \,|\, \{j, k\}\rangle$, and $\langle k, l \,|\, \{i, j\}\rangle$ are in $\mathcal{J}^{H_3}$. It can be seen that none of semi-graphoid axioms or the composition property can be used to imply $\langle l, \{i, k\} \,|\, j\rangle \in \mathcal{J}^{H_3}$. The intersection property is the only axiom that implies the result. See also e.g. Example 3.26 of Lauritzen [1996], showing that the pairwise Markov property does not imply the global Markov property for DAGs when intersection is violated.

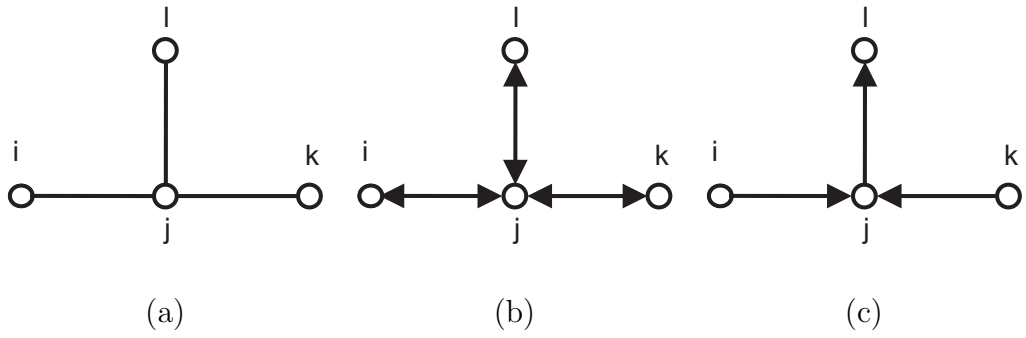


Figure 2.17: For the equivalence of pairwise and global Markov properties, (a) an undirected graph H_1 that shows that the intersection property is necessary; (b) a bidirected graph H_2 that shows that the composition property is necessary; (c) an directed acyclic graph H_3 that shows that the intersection property is necessary.

It is known that, for UGs, the five graphoid axioms are necessary and sufficient for the equivalence of pairwise and global Markov properties; see Lauritzen [1996]. For BGs, in fact, only the five compositional semi-graphoid axioms are necessary. This can be inferred from the proof of Theorem 2.3, since part I of the proof is not relevant for bidirected graphs unless $C = \varnothing$, and the intersection property is

not used in part II of the proof.

Markov properties for probability distributions. As described in Section 1.3.2, here we consider conditional independence in the special situation of having a collection of random variables from some probability distributions.

We know that probability distributions induce probabilistic independence models, hence pairwise and global Markov properties can be defined for probabilistic independence models that conform with the graph.

Following the discussion of probability distributions and compositional graphoid axioms in Chapter 1, an important corollary of Theorem 2.3 is then given as follows:

Corollary 2.3. *Let H be a maximal loopless mixed graph. A probabilistic independence model that satisfies the intersection and composition axioms satisfies the pairwise Markov property if and only if it satisfies the global Markov property.*

Proof. The induced independence models of all probability distributions satisfy the semigraphoid axioms. If the induced model satisfies intersection and composition properties then it is, in addition, a compositional graphoid. Thus, Theorem 2.3 implies the result. □

This corollary explains the importance of the family of Gaussian distributions in graphical models, since it is one of the few known families of distributions whose induced independence models satisfy both intersection and composition properties. Another known family of distributions whose induced independence model satisfies these properties is the family of distributions defined for symmetric binary variables, introduced by Wermuth et al. [2009]. The seemingly surprising result in the paper, regarding the regression model for these variables, is a consequence of the composition property, satisfied by the distribution.

2.6 Summary and future work

Summary. The most important contribution of this chapter is the equivalence of the pairwise Markov property and the global Markov property with respect to m-separation for maximal RGs when the compositional graphoid axioms hold. Such an equivalence is essential as it is very common to generate graphs by using the implied pairwise independence statements from sets of random variables with joint probability distributions. This result has notably shown the importance of the composition property that has been mostly neglected in the literature.

From the subclasses of LMGs, RGs and its subclasses are of special interest since, as will be seen in the next chapter, these preserve marginal and conditional independence models over a DAG.

Future work. As has been outlined in Section 2.3, the important class of chain graphs with the LWF Markov property is not included in the class of LMGs with m-separation. However, the existence of a direct path-based separation criterion, called c-separation [Studeny and Bouckaert, 1998, Studeny, 1998], allows us to include such a class in the class of LMGs with the c-separation criterion, and to apply the same methodology to such a class to prove that the global Markov property with respect to the c-separation criterion is equivalent to a defined pairwise Markov property when graphoid or similar axioms hold.

The class of LMGs with c-separation will not contain the class of RGs, but it will share with LMGs with m-separation the class of AAGs and its subclasses as special cases. This is a consequence of the lack of armor in such graphs. Details of this will be investigated in the future.

Chapter 3

Stable mixed graphs

3.1 Marginalisation, conditioning, and stability

Marginal and conditional independence models. Consider an independence model $\mathcal{J}$ over a set V. For M a subset of V, the *independence model $\mathcal{J}$ after marginalisation over M*, denoted by $\alpha(\mathcal{J}; M, \varnothing)$, is the subset of $\mathcal{J}$ whose triples do not contain members of M, i.e.

$$\alpha(\mathcal{J}; M, \varnothing) = \{\langle A, B \mid D\rangle \in \mathcal{J} : (A \cup B \cup D) \cap M = \varnothing\}.$$

One can observe that $\alpha(\mathcal{J}; M, \varnothing)$ is an independence model over $V \setminus M$.

For C a subset of V, the *independence model after conditioning on C*, denoted by $\alpha(\mathcal{J}; \varnothing, C)$, is

$$\alpha(\mathcal{J}; \varnothing, C) = \{\langle A, B \mid D\rangle : \langle A, B \mid D \cup C\rangle \in \mathcal{J} \text{ and } (A \cup B \cup D) \cap C = \varnothing\}.$$

One can also observe that $\alpha(\mathcal{J}; \varnothing, C)$ is an independence model over $V \setminus C$.

Combining these definitions, for disjoint subsets M and C of V, the *independence model after marginalisation over M and conditioning on C* is

$$\alpha(\mathcal{J}; M, C) = \{\langle A, B \mid D\rangle : \langle A, B \mid D \cup C\rangle \in \mathcal{J} \text{ and } (A \cup B \cup D) \cap (M \cup C) = \varnothing\},$$

which is an independence model over $V \setminus (M \cup C)$.

Notice here that α is a function from the set of independence models and two of

their subsets to the set of independence models. Notice also that operations for marginalisation and conditioning commute.

Marginal and conditional probability distributions. Marginalisation and conditioning in probability conform with marginalisation and conditioning for independence models. Once again, consider C a subset of the node set V of a graph G and M another disjoint subset of V. Knowing the joint density function $f_V = f_{X_V}$ over G, the *marginal density* of V over M is

$$f_{V\setminus M}(x_{V\setminus M}) = \int_{-\infty}^{\infty} f_V(x_V) dx_M.$$

In the discrete case summation is used instead of integration to find the *marginal frequency function*. Similarly, the *conditional density* of V over C is

$$f_{V\,|\,C}(x_V \,|\, x_C) = \frac{f_V(x_V)}{f_C(x_C)}.$$

It is easy to deduce that $f_{(V\setminus M)\,|\,C} = f_{(V\,|\,C)\setminus M}$. We call this the *marginal and conditional density* of V over M and C.

Now consider a set $N = V \setminus (M \cup C)$ and a collection of random variables $(X_\alpha)_{\alpha\in N}$ with joint density $f_{(V\setminus M)\,|\,C}$. We associate an independence model to this density as outlined in Section 1.3.2. It can be shown that if $\mathcal{J}$ is the associated independence model to the collection of random variables $(X_\alpha)_{\alpha\in V}$ with joint density f_V then the associated independence model to $(X_\alpha)_{\alpha\in N}$ with joint density $f_{(V\setminus M)\,|\,C}$ is $\alpha(\mathcal{J}, M, C)$.

Stability under marginalisation and conditioning. Consider a family of graphs $\mathcal{T}$ with a family of independence models $\mathcal{J}^{\mathcal{T}} = \{\mathcal{J}^G\}_{G\in\mathcal{T}}$, defined over the node sets of members of $\mathcal{T}$. We call $\mathcal{T}$ *stable* (under marginalisation and conditioning) with respect to $\mathcal{J}^{\mathcal{T}}$ if, for every graph $G = (V, E) \in \mathcal{T}$ and every disjoint subsets M and C of V, there is a graph $H \in \mathcal{T}$ such that $\mathcal{J}^H = \alpha(\mathcal{J}^G; M, C)$.

If there is a graph $H \in \mathcal{T}$ such that $\mathcal{J}^H = \alpha(\mathcal{J}^G; \varnothing, C)$ then $\mathcal{T}$ is stable un-

der conditioning with respect to $\mathcal{J}^{\mathcal{T}}$, and if there is a graph $H \in \mathcal{T}$ such that $\mathcal{J}^H = \alpha(\mathcal{J}^G; M, \varnothing)$ then $\mathcal{T}$ is stable under marginalisation with respect to $\mathcal{J}^{\mathcal{T}}$.

Notice that if the node set of such a graph H is N then $N = V \setminus (M \cup C)$.

A note on the names and notations for special types of LMGs. Notice that in the literature there is often no distinction between the name of the type of a graph and the name of the model in which the graph is used. Henceforth, we denote the subclasses of LMGs by the name of the type of graphical models that these are used to describe. The names defined in Chapter 2 were based solely on the graphical characteristics of graphs, whereas the names in the literature, which we use henceforth, are often based on the statistical uses of the models these induce. Particularly, we use "summary graph (SG)" for "AAG", and "multivariate regression chain graph (MRCG)" for "BCG".

We also use prefix "M" as an abbreviation for "maximal". For example, "MAG" stands for "maximal AG", "MSG" for "maximal SG", and so on.

In addition, we use the notations $\mathcal{RG}$, $\mathcal{SG}$, $\mathcal{AG}$, $\mathcal{ADMG}$, $\mathcal{MRCG}$, $\mathcal{UG}$, $\mathcal{BG}$, and $\mathcal{DAG}$ for the set of all RGs, SGs, AGs, ADMGs, MRCGs, UGs, BGs, and DAGs respectively.

Stability of subclasses of LMGs. Using the m-separation criterion, we investigate which subclasses of LMGs are stable. We will show that, among the subclasses of LMGs that we discussed in Chapter 2, RGs, SGs, AGs, UGs, and BGs are stable.

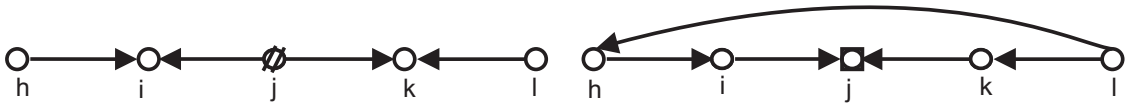
On the other hand, DAGs, MRCGs, and ADMGs are not stable. G_1 in Figure 3.1(a) shows that DAGs are not stable with respect to $\mathcal{J}_m(\mathcal{DAG})$:

For DAG G_1, $\langle h, k \,|\, \varnothing \rangle$ and $\langle l, i \,|\, \varnothing \rangle$ are in $\mathcal{J}_m(G_1)$, but $\langle h, k \,|\, i \rangle$ and $\langle l, i \,|\, k \rangle$ are not in $\mathcal{J}_m(G_1)$. This implies that the first two statements are in $\alpha(\mathcal{J}_m(G_1); j, \varnothing)$

and the last two statements are not in $\alpha(\mathcal{J}_m(G_1); j, \varnothing)$.

Suppose, for contradiction, that there is a DAG H such that $\mathcal{J}_m(H) = \alpha(\mathcal{J}_m(G_1); j, \varnothing)$. We know that the node set of H is $\{h, i, k, l\}$. Statement $\langle l, i \,|\, \varnothing\rangle \in \mathcal{J}_m(H)$ implies that $l \not\sim i$ in H. Statement $\langle l, i \,|\, k\rangle \notin \mathcal{J}_m(H)$ implies that there is an m-connecting path between l and i given k. If the path is $\langle l, h, i\rangle$ then $\langle l, i \,|\, \varnothing\rangle \in \mathcal{J}_m(H)$ does not hold. In addition, $k \not\sim h$ because of $\langle h, k \,|\, \varnothing\rangle \in \mathcal{J}_m(H)$. Hence $\langle i, k, l\rangle$ is the m-connecting path and therefore there is a collider V-configuration $\langle i, k, l\rangle$ in H.

By the same argument we imply that there is a collider V-configuration $\langle h, i, k\rangle$ in H. This is a contradiction since a DAG cannot contain both collider V-configurations $\langle h, i, k\rangle$ and $\langle i, k, l\rangle$. Therefore, DAGs are not stable.



(a) (b)

Figure 3.1: (a) A directed acyclic graph G_1, which shows DAGs are not stable under marginalisation. ($\not\emptyset \in M$.) (b) A directed acyclic graph G_2, which shows DAGs, MRCGs and ADMGs are not stable under conditioning. ($\boxed{\bigcirc} \in C$.)

MRCG (and ADMG) G_2 in Figure 3.1(b) shows that MRCGs and ADMGs are not stable with respect to $\mathcal{J}_m(\mathcal{MRCG})$ and $\mathcal{J}_m(\mathcal{ADMG})$ respectively:

For G_2, $\langle h, k \,|\, j\rangle$ and $\langle l, i \,|\, j\rangle$ are not in $\mathcal{J}_m(G_2)$, but $\langle h, k \,|\, \{i, l, j\}\rangle$ and $\langle l, i \,|\, \{k, h, j\}\rangle$ are in $\mathcal{J}_m(G_2)$. This implies that $\langle h, k \,|\, \varnothing\rangle$ and $\langle l, i \,|\, \varnothing\rangle$ are not in $\alpha(\mathcal{J}_m(G_2); \varnothing, j)$ and $\langle h, k \,|\, \{i, l\}\rangle$ and $\langle l, i \,|\, \{k, h\}\rangle$ are in $\alpha(\mathcal{J}_m(G_2); \varnothing, j)$. In addition, $\langle i, k \,|\, j\rangle \notin \mathcal{J}_m(G_2)$ and $\langle i, k \,|\, \{l, h, j\}\rangle \notin \mathcal{J}_m(G_2)$ imply that $\langle i, k \,|\, \varnothing\rangle \notin \alpha(\mathcal{J}_m(G_2); \varnothing, j)$ and $\langle i, k \,|\, \{l, h\}\rangle \notin \alpha(\mathcal{J}_m(G_2); \varnothing, j)$.

Suppose, for contradiction, that there is an ADMG H such that $\mathcal{J}_m(H) =$

$\alpha(\mathcal{J}_m(G_2); \varnothing, j)$. We know that the node set of H is $\{h, i, k, l\}$. In addition, by Corollary 2.2, we can assume that H is maximal. This implies that in H there are hi-, kl-, and hl-edges since there is no independence statement of form $\langle h, i \,|\, S\rangle$, for some S, in $\mathcal{J}_m(H)$, etc. In addition, $i \sim k$ since if $i \not\sim k$ then $\langle i, k \,|\, \varnothing\rangle \notin \alpha(\mathcal{J}_m(G_2); \varnothing, j)$ does not hold in the case that $\langle k, l, h\rangle$ is collider and $\langle i, k \,|\, \{l, h\}\rangle \notin \alpha(\mathcal{J}_m(G_2); \varnothing, j)$ does not hold in the case that $\langle k, l, h\rangle$ is non-collider. Moreover, $k \not\sim h$ and $l \not\sim i$ because $\langle h, k \,|\, i\rangle \in \mathcal{J}_m(H)$ and $\langle l, i \,|\, \{k, h\}\rangle \in \mathcal{J}_m(H)$. Therefore H is a cycle $\langle i, k, l, h\rangle$.

Now $\langle h, k \,|\, \{i, l\}\rangle \in \alpha(\mathcal{J}_m(G_2); \varnothing, j)$ implies that $\langle k, i, h\rangle$ and $\langle k, l, h\rangle$ are non-collider and $\langle l, i \,|\, \{k, h\}\rangle \in \alpha(\mathcal{J}_m(G_2); \varnothing, j)$ implies that $\langle i, k, l\rangle$ and $\langle i, h, l\rangle$ are non-collider. However, if all the edges are arrows or arcs it is easy to see that there is a collider V-configuration in H unless $\langle i, k, l, h\rangle$ is a direction-preserving cycle, a contradiction. Therefore, ADMGs, MRCGs, and DAGs are not stable.

Since henceforth we only work with the m-separation criterion, we usually neglect mentioning that stability is with respect to $\mathcal{J}_m(\mathcal{T})$.

Stable mixed graphs. We showed that DAGs are not stable. We look for stable classes of graphs that include the class of DAGs as a subclass. In this chapter we discuss three known types of such graphs, namely RGs (as a modification of MC graphs as has been outlined in Section 2.2), SGs, and AGs. All these use mixed graphs together with the m-separation criterion. We specifically call these graphs *stable mixed graphs*, and we will see that in these graphs arcs are related to marginalisation and lines are related to conditioning.

Throughout the thesis we usually use the notation $G = (V, E)$ to represent a DAG and $H = (N, F)$ to represent a stable mixed graph, where $N = V \setminus (M \cup C)$ and M and C are the subsets of V that we marginalise over and condition on respectively. In addition, for the graph $H_2 \in \mathcal{T}$ for which $\mathcal{J}^{H_2} = \alpha(\mathcal{J}_m(H_1); M, C)$,

we use the notation $H_2 = \alpha_{\mathcal{T}}(H_1; M, C)$. For each type of stable mixed graphs, we later precisely define $\alpha_{\mathcal{T}}$ with specific algorithms. We call $\alpha_{\mathcal{T}}$ a *generating function* or more specifically a *$\mathcal{T}$-generating function*.

Structure of the chapter. Each of the next three sections of this chapter deals with one type of stable mixed graphs. We discuss RGs in Section 2, SGs in Section 3, and AGs in Section 4. In each section we introduce a theoretically straightforward algorithm to generate stable mixed graphs from DAGs or from graphs of the same class, as well as an equivalent algorithm that is local in the sense that it looks solely for V-configurations in graph (after determining the ancestor set). For each type of stable mixed graph, we prove that the graphs and algorithms are well-defined in the sense that instead of marginalising over or conditioning on a set of nodes, by splitting the marginalisation or conditioning set into two subsets, one can marginalise over or condition on the first subset first, and then marginalise over or condition on the second subset and obtain an equal graph. We also prove that the generated graphs induce the modified independence model after marginalisation and conditioning.

Despite the similarities between the three types of stable mixed graphs there is no obvious way of relating them to one another. The algorithms that are introduced in this chapter define generating functions, which will be used in the last section to better understand the exact relationship between different types of stable mixed graphs.

3.2 Ribbonless graphs

In this thesis we do not explain how to generate an MC graph from a DAG by marginalisation and conditioning; see Koster [2002] for the process. Instead, we deal with the RGs as a modification of MC graphs, as was outlined in Section 2.2.

This is because the relationship between MC graphs and RGs has already been established, and RGs have a similar structure and independence interpretation to other types of stable mixed graphs.

Goal and structure of the section. Here we define two algorithms to generate an RG from a given RG and two subsets of its node set that will be marginalised over and conditioned on. Obviously these algorithms can also accept DAGs as an input since DAGs are a subclass of RGs. The first algorithm is global in the sense that it looks for paths of a specific type in the graph. The second algorithm is local in the sense that it looks solely for V-configurations in the graph (after determining the ancestor set). The second algorithm is more suitable for computer implementation and is computationally polynomial in the number of nodes and edges of the graph.

Later in this section we will show that these two algorithms generate the same RG. We treat these algorithms as a function and prove that the function is well-defined and the generated graph induces the independence model of the input graph after marginalisation and conditioning.

3.2.1 Generating ribbonless graphs

A global algorithm to generate RGs from RGs. Suppose that H is an RG and consider M and C two disjoint subsets of the node set. To introduce the algorithm we use a generalisation of m-connecting paths given C. A path is said to be *m-connecting given M and C* if along it every collider node is in $C \cup \mathrm{an}(C)$ and every non-collider node is in M.

Algorithm 3.1. $\alpha'_{RG}(H; M, C)$*: (Generating an RG from RG* $H = (N, F)$*)*
Start from H*. For every* i *and* j *in* $N \setminus (M \cup C)$ *connected by an* m*-connecting path* π *given* M *and* C *in* H*, generate an endpoint-identical edge to* π *between* i *and* j *if the edge does not already exist, i.e.*

1. *generate an arrow from j to i if on π there is no arrowhead pointing to j but there is an arrowhead pointing to i and if the arrow does not already exist;*

2. *generate an ij-arc if on π there are arrowheads pointing to both endpoints and if the arc does not already exist;*

3. *generate an ij-line if on π there are no arrowheads pointing to endpoints and if the line does not already exist.*

Then remove all nodes in $M \cup C$.

Figure 3.2 illustrates how to apply Algorithm 3.1 to a DAG.

(a) (b)

Figure 3.2: (a) A directed acyclic graph G, $\not\emptyset \in M$ and $\boxed{\text{O}} \in C$. (b) The generated RG from G after applying step 1.

The map $\alpha'_{RG}(H; M, C)$. We will prove that α'_{RG} generates RGs. We consider α' to be a function from the set of RGs and two subsets of their node set to the set of RGs.

A local algorithm to generate RGs from RGs. After determining the ancestors of the nodes that are conditioned on in a graph, instead of looking for all m-connecting paths in a graph, by a local algorithm, one can only look for V-configurations with inner nodes in $M \cup C \cup \text{an}(C)$ to obtain the same graph. There are 10 possible non-isomorphic V-configurations in an RG, displayed in Table 3.1. We now define the following algorithm, derived from Wermuth et al. [1994], Koster [2002], and Wermuth [2011](appendix), for locally generating an RG from an RG after determining the ancestor set:

Algorithm 3.2. $\alpha_{RG}(H; M, C)$: *(Generating an RG from a ribbonless graph H) Start from H.*

Generate an endpoint-identical edge between the endpoints of collider V*s with inner node in $C \cup \mathrm{an}(C)$ and non-collider* V*s with inner node in M, i.e. generate an appropriate edge as in Table 3.1 between the endpoints of every V-configuration with inner node in M or $C \cup \mathrm{an}(C)$ if the edge of the same type does not already exist.*

Apply the previous step until no other edge can be generated. Then remove all nodes in $M \cup C$.

Table 3.1: Types of edge induced by V-configurations with inner node in $m \in M$ or $s \in C \cup \mathrm{an}(C)$.

1	i $\prec$—m $\prec$— j	generates	i $\prec$—j
2	i $\prec$—m —— j	generates	i $\prec$—j
3	i $\prec\succ$m —— j	generates	i $\prec$—j
4	i $\prec$—m —$\succ$ j	generates	i $\prec\succ$j
5	i $\prec$—m $\prec\succ$ j	generates	i $\prec\succ$j
6	i ——m $\prec$— j	generates	i ——j
7	i ——m —— j	generates	i ——j
8	i $\prec\succ$s $\prec$— j	generates	i $\prec$—j
9	i $\prec\succ$s $\prec\succ$ j	generates	i $\prec\succ$j
10	i —$\succ$s $\prec$— j	generates	i ——j

This method is a generalisation of the method used by Lauritzen et al. [1990], called *moralisation*, as a separation criterion on DAGs. It is seen that the table generates endpoint-identical edges to the V-configurations.

Lemma 3.1. *The order of applying steps of Table 3.1 in Algorithm 3.2 is irrelevant.*

Proof. The result follows from the fact that adding an edge between endpoints of a V-configuration does not destroy other V-configurations or direction-preserving paths in the graph; thus at each iteration of the algorithm if there are two V-configurations then it does not matter which one to be chosen first. □

Figure 3.3 illustrates how to apply Algorithm 3.2 step by step to the DAG in Figure 3.2. We start from step 1 of Table 3.1 and proceed step by step. We return to step 1 at the end if there are any applicable steps left. As will be observed this algorithm generates the same graph as that in Algorithm 3.1. We prove this for the general case after this example.

We will show that α_{RG} is an RG-generating function. Since $\mathcal{DAG} \subset \mathcal{RG}$, one can also use Algorithm 3.2 to generate an RG from a DAG. Notice that it is not enough to simply apply steps 1, 4, and 10 of Table 3.1 to a DAG; for example see the graph in Figure 3.2.

Computational complexity of the local algorithm. Regarding computational complexity, Algorithm 3.2 is polynomial. For example, one can find $C \cup \mathrm{an}(C)$ in $|C \cup \mathrm{an}(C)|^2 e$ steps by the following procedure, where e is the number of edges of the graph: For each member i of the list of nodes in C, by (1) going through all the edges; and (2) adding parents of i at the end of the list (if these are not already in the list).

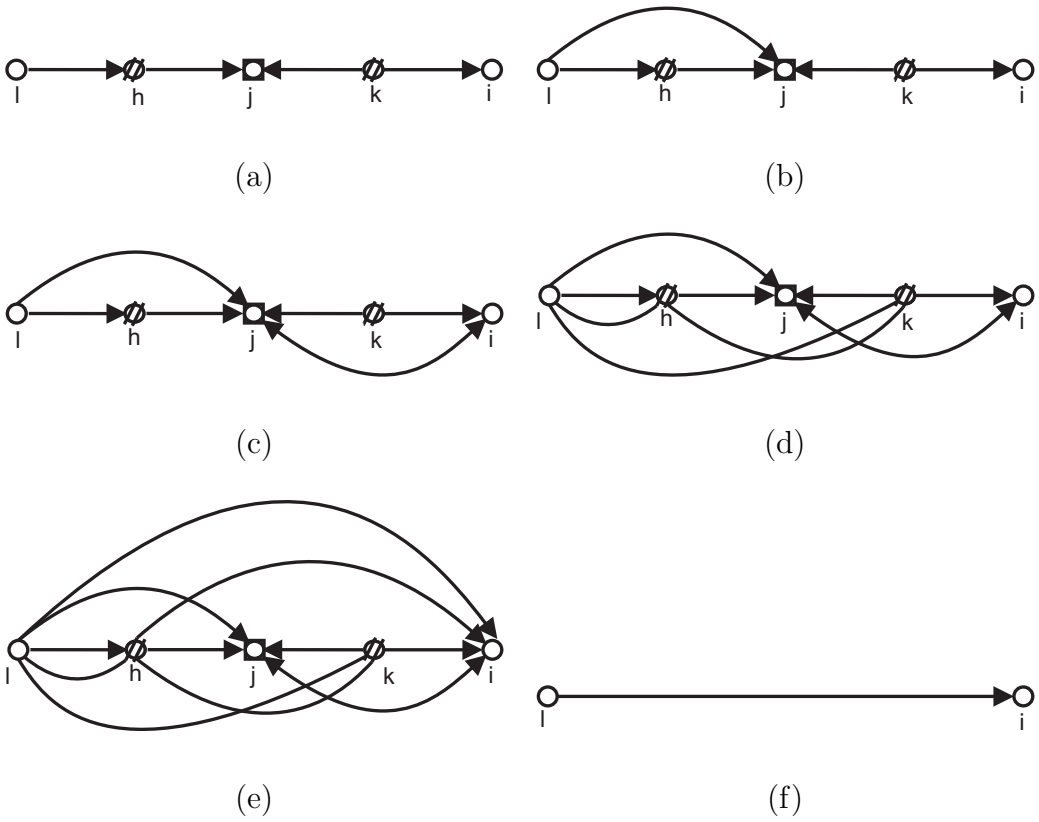


Figure 3.3: (a) A directed acyclic graph G, $\not\emptyset \in M$ and $\boxed{\bigcirc} \in C$. (b) The generated graph after applying step 1 of the table. (c) The generated graph after applying step 4. (d) The generated graph after applying step 10. (e) The generated graph after applying step 8. (f) The generated RG from G.

In order to generate edges, for each member i of M or $C \cup \mathrm{an}(C)$ one looks for all subsets of the edge set of the generated graph that are of length two and contain i to deduce whether a new edge should be generated. By going through the edge set once more, one can deduce whether the generated edge already exists. This process can be done in fewer steps than $(|M| + |C \cup \mathrm{an}(C)|)\binom{f}{2}f$, where f is the number of edges of the generated graph.

Removing the nodes in M and C can be easily done in $(|M| + |C|)f$ steps. Therefore, in the worst case scenario one can apply the algorithm in $\binom{n^2}{2}n^2(|M| + |C \cup \mathrm{an}(C)|)$ steps and the computational complexity is $O(n^7)$.

Some widely used lemmas. Here we introduce a lemma that is widely used in our proofs, both for RGs and for other types of stable mixed graphs.

Lemma 3.2. *Let H be an RG. If there exists an edge between i and j in $\alpha_{RG}(H; M, C)$ then between i and j in H there exists an endpoint-identical m-connecting path given M and C to the ij-edge.*

Proof. If an edge between i and j in $\alpha_{RG}(H; M, C)$ does not exist in H then it has been generated by certain intermediate graphs that have each been generated by adding one edge to the previous graph by one of the steps of Table 3.1. We denote these graphs by the sequence $\langle H = H_0, H_1, \ldots, H_n, \alpha_{RG}(H; M, C)\rangle$, where H_n is the last step before removing M and C.

We prove by reverse induction on p that in all H_p, $0 \leq p \leq n$, between i and j there exists a path on which non-collider inner nodes are in M and collider inner nodes or their descendants are either in C or an endpoint of a line. For $p = n$, there is obviously an edge between i and j. We show that if there is such a path in H_r then we can find the same type of path between i and j in H_{r-1}.

If all edges along the path exist in H_{r-1} then we should check that a collider node that is an ancestor of a member of C or an ancestor of a node that is an endpoint of a line in H_r is an ancestor of a member of C or an ancestor of a node that is an endpoint of a line in H_{r-1}. If an arrow has been generated along the direction-preserving path in H_r then it has been generated by the V-configuration $\langle i', m, j'\rangle$ of the first three steps or the V-configuration $\langle i', s, j'\rangle$ of step 8 of Table 3.1. If it is step 1 then we can replace the $i'j'$-arrow by $\langle i', m, j'\rangle$ to obtain a direction-preserving path. If it is steps 2 or 3 then node j' is an endpoint of a line and we are done. If it is step 8 then, since $s \in C \cup \mathrm{an}(C)$, the inner node of the V-configuration is in $\mathrm{an}(C)$ and we are done.

Thus suppose that an $i'j'$-edge along the m-connecting path is the edge that has been generated by this step. This has been generated by one of V-configurations

of Table 3.1. Since in all cases the V-configuration is endpoint-identical to the $i'j'$-edge, and since all inner nodes of the non-collider V-configurations are in M and all inner nodes of the collider V-configurations are in $C \cup \mathrm{an}(C)$, by placing the V-configuration instead of the $i'j'$-edge on the path, we still get a path whose non-collider inner nodes are in M and either whose collider inner nodes are in $C \cup \mathrm{an}(C)$ or whose collider nodes or a descendant of them are an endpoint of a line, as required.

Therefore, by reverse induction, there exists a path, as described above, in H. However, since H is ribbonless, the path cannot contain a collider V-configuration $\langle i', h, j' \rangle$ such that h or a descendant of h is an endpoint of a line unless $i' \sim j'$ and the $i'j'$-edge is endpoint-identical to $\langle i', h, j' \rangle$. In this case, the $i'j'$-edge can be used instead of $\langle i', h, j' \rangle$ and, by induction, we obtain an m-connecting path given M and C between i and j.

The fact that in Table 3.1 the V-configurations are endpoint-identical to the generated $i'j'$-edges implies that all discussed paths in each H_p are endpoint-identical. □

This lemma illustrates that the map α_{RG} generates endpoint-identical edges to m-connecting paths. The two following lemmas illustrate what happens to a direction-preserving path after applying the algorithm.

Lemma 3.3. *If $i \in \mathrm{an}(j)$ in $\alpha_{RG}(H; M, C)$ then in H one of the following holds: 1) $i \in \mathrm{an}(j)$; 2) i or a descendant of i is an endpoint of a line; 3) $i \in \mathrm{an}(C)$.*

Proof. We know that there is a direction-preserving path $\pi = \langle i = i_0, i_1, \ldots, i_p = j \rangle$ in $\alpha_{RG}(H; M, C)$. Consider the $i_0 i_1$-edge. By Lemma 3.2 in H given M and C there is an m-connecting path between i_0 and i_1, on which there is no arrowhead pointing to i_0. It is easy to observe that if this path is not a direction-preserving path then one of the following holds: (1) i_0 is an ancestor of a collider node on the path, which is in $C \cup \mathrm{an}(C)$. This implies that $i \in \mathrm{an}(C)$; (2) i_0 is an endpoint

of a line or an ancestor of a node that is an endpoint of a line on the path. If (1) or (2) holds then we are done, hence assume that $i_0 \in \text{an}(i_1)$. By the same argument and by induction along nodes of π we conclude the result. □

Lemma 3.4. *For i and j outside $M \cup C$, if $i \in \text{an}(j)$ in H then one of the following holds: 1) $i \in \text{an}(j)$ in $\alpha_{RG}(H; M, C)$; 2) $i \in \text{an}(C)$ in H.*

Proof. We know that there is a direction-preserving path $\pi = \langle i = i_0, i_1, \ldots, i_p = j \rangle$ in H. Consider the $i_0 i_1$-edge. We now have three cases: 1) If $i_1 \in C$ then $i \in \text{an}(C)$ in H and we are done. 2) If $i_1 \in M$ then Algorithm 3.2 generates an arrow from i_0 to i_2. 3) If $i_1 \notin M \cup C$ then $i_0 \in \text{an}(i_2)$ in $\alpha_{RG}(H; M, C)$. By the same argument and by induction along the nodes of π we conclude the result. □

Here we state a similar result to the result of Lemma 2.11 for the combination of m-connecting paths given M and C.

Lemma 3.5. *In an RG suppose that given M and C there are m-connecting paths $\pi_1 = \langle i = i_0, i_1, \ldots, i_n, h \rangle$ between i and h and $\pi_2 = \langle j = j_0, j_1, \ldots, j_m, h \rangle$ between h and j. The combination $\pi_{12} = \pi_1 \circ \pi_2$ is then an m-connecting path between i and j given M and C in each of the following situations:*

a1) *$\langle i_n, h, j_m \rangle$ is collider and $h \in C \cup \text{an}(C)$;*

a2) *$i_n = j_m$ with an arrowhead pointing to h on the $i_n h$-edge and $h \in C \cup \text{an}(C)$;*

b1) *$\langle i_n, h, j_m \rangle$ is non-collider and $h \in M$;*

b2) *$i_n = j_m$ with no arrowhead pointing to h on the $i_n h$-edge and $h \in M$;*

c1) *$\langle i_n, h, j_m \rangle$ is collider and h or a descendant of h is an endpoint of a line or on a direction-preserving cycle;*

c2) *$i_n = j_m$ with an arrowhead pointing to h on the $i_n h$-edge and h or a descendant of h is an endpoint of a line or on a direction-preserving cycle.*

Proof. Let $\pi_{12} = \pi_1 \circ \pi_2 = \langle i, \ldots, k, j_{q-1}, \ldots, j \rangle$ be the combination of π_1 and π_2. If $k = h$ and either a1) or b1) holds, the conclusion is obvious. If c1) holds then $\langle i_n, h, j_m \rangle$ is a ribbon unless there is an $i_n j_m$-edge that is endpoint-identical to the V-configuration. This edge can be used instead of $\langle i_n, h, j_m \rangle$ to establish an m-connecting path. The cases a2), b2), or c2) are only relevant when $k \neq h$.

Next consider the situation where $k \neq h$. Since π_1 and π_2 are m-connecting, for π_{12} to be m-connecting we only need to check the V-configuration $\langle i_{p-1}, k, j_{q-1} \rangle$. We have to deal with two cases:

Case 1: $\langle i_{p-1}, k, j_{q-1} \rangle$ is a non-collider. In this case there is no arrowhead pointing to k from at least one of i_{p-1} or j_{q-1}. This means that $\langle i_{p-1}, k, i_{p+1} \rangle$ on π_1 or $\langle j_{q-1}, k, j_{q+1} \rangle$ on π_2 is a non-collider, and since π_1 and π_2 were both m-connecting we have $k \in M$. Hence π_{12} is m-connecting.

Case 2: $\langle i_{p-1}, k, j_{q-1} \rangle$ is a collider. We need to consider the following two subcases:

Case 2.1. If $\langle i_{p-1}, k, j_{q-1} \rangle$ is a collider and any of $\langle i_{p-1}, k, i_{p+1} \rangle$ or $\langle j_{q-1}, k, j_{q+1} \rangle$ is also a collider then $k \in C \cup \mathrm{an}(C)$ and π_{12} is m-connecting.

Case 2.2. If $\langle i_{p-1}, k, j_{q-1} \rangle$ is a collider but $\langle i_{p-1}, k, i_{p+1} \rangle$ and $\langle j_{q-1}, k, j_{q+1} \rangle$ are both non-colliders then by Lemma 2.6, the subpath of π_1 between k and a collider node l_1 or h is an anterior path and similarly for π_2, l_2, and h. Now we have the three following further cases:

Case 2.2.1: $k \in \mathrm{ant}(l_1) \cup \mathrm{ant}(l_2) \cup \mathrm{ant}(h) \setminus (\mathrm{an}(l_1) \cup \mathrm{an}(l_2) \cup \mathrm{an}(h))$. In this case, by Lemma 2.2, $\langle i_{p-1}, k, j_{q-1} \rangle$ is a ribbon unless there is an endpoint-identical $i_{p-1} j_{q-1}$-edge in the graph, which can be used instead of $\langle i_{p-1}, k, j_{q-1} \rangle$ on π_{12} to generate an m-connecting path.

Case 2.2.2: One of the subpaths of π_1, π_2 from k to l_1, l_2 is direction-preserving. Because π_1 and π_2 are m-connecting we must have l_1 or l_2 in $C \cup \mathrm{an}(C)$. Thus $k \in \mathrm{an}(C)$ and π_{12} is m-connecting.

Case 2.2.3: Both subpaths of π_1 and π_2 from k to h are direction-preserving. Then $\langle i_n, h, j_m \rangle$ is collider or $i_n = j_m$ with an arrowhead pointing to h on the $i_n h$-edge. Thus if a1) or a2) holds π_{12} is m-connecting since then $h \in C \cup \mathrm{an}(C)$. If c1) or c2) holds then $\langle i_{p-1}, k, j_{q-1} \rangle$ is a ribbon unless there is an $i_{p-1} j_{q-1}$-edge that is endpoint-identical to the V-configuration. This edge can be used instead of $\langle i_{p-1}, k, j_{q-1} \rangle$ to establish an m-connecting path. In addition, if b1) or b2) holds this case is not possible. □

Equivalence of α_{RG} and α'_{RG}. Here we show that the two defined algorithms are equivalent.

Proposition 3.1. *Let $H = (N, F)$ be a ribbonless graph. Algorithms 3.1 and 3.2 are equivalent, i.e. for disjoint subsets M and C of N, $\alpha_{RG}(H; M, C) = \alpha'_{RG}(H; M, C)$.*

Proof. Suppose that the sequence $\langle H = H_0, H_1, \ldots, H_n, \alpha_{RG}(H; M, C) \rangle$ consists of intermediate graphs that have each been generated by adding one edge to the previous graph, where H_n is the last step before removing M and C. Because of Lemma 3.1 we can determine the order of the intermediate graphs while the final generated graph is the same in all possible orderings.

($\Rightarrow$) Suppose that there is one of the V-configurations $\langle i, m, j \rangle$ (or $\langle i, s, j \rangle$) of Table 3.1 in H_p, $0 \leq p \leq n$. By Lemma 3.2 in H, there are m-connecting paths given M and C between i and m (or s) and between m (or s) and j such that the m-connecting paths are endpoint-identical to the two edges of the V-configuration. Since $m \in M$ and $s \in C \cup \mathrm{an}(C)$, by Lemma 3.5 there is an m-connecting path given M and C between i and j in H. Now using algorithm

3.1 implies the result.

($\Leftarrow$) Conversely, we show that three steps of Algorithm 3.1 can be implied by Algorithm 3.2, using the method outlined in Table 3.1.

Suppose that there is an m-connecting path π given M and C between i and j in H_k, $0 \leq k \leq n-1$. We prove as long as $r > 0$ if there is an m-connecting path given M and C between i and j in H_k with r inner nodes then there is an m-connecting path given M and C between i and j in H_{k+1} with $r-1$ inner nodes. By induction we will finally get an m-connecting path between i and j without inner nodes, i.e. an edge between i and j.

Consider an m-connecting path given M and C between i and j in H_k with $r > 0$ inner nodes. Consider an arbitrary inner node on the path. If this node is a collider node then one of the V-configurations 8, 9, or 10 of Table 3.1 is employed to generate an edge between the endpoints of the V-configuration in H_{k+1}. Since the generated edge is endpoint-identical to the V-configuration, one can use the generated edge instead of the V-configuration to obtain an m-connecting path with $r-1$ inner nodes. If the arbitrary node is non-collider then one of the other V-configurations of Table 3.1 is used.

It is easy to check that the generated edges are endpoint-identical to the m-connecting paths in the final graph. This implies the result for each of the three steps. □

Basic properties of α_{RG}. Here we introduce the basic properties that α_{RG} must have to be able to act as a generating function.

Proposition 3.2. *Graphs generated by Algorithms 3.1 and 3.2 are RGs.*

Proof. Because of Proposition 3.1, here we only work with Algorithm 3.1. Graphs generated by Algorithm 3.1 have trivially three desired types of edges and are loopless.

Now suppose, for contradiction, that there is a ribbon $\langle i, h, j\rangle$ in a generated graph $\alpha_{RG}(H; M, C)$. By Lemma 3.2 in H given M and C there are m-connecting paths $\pi_1 = \langle i = i_0, i_1, \ldots, i_n, h\rangle$ between i and h and $\pi_2 = \langle j = j_0, j_1, \ldots, j_m, h\rangle$ between h and j such that there are arrowheads at h on both $i_n h$- and $j_m h$-edges. (Notice that it is possible that $i_n = j_m$ and it is also possible that $i_n = i$ or $j_m = j$ in H.)

We also know that, in $\alpha_{RG}(H; M, C)$, the node h is an endpoint of a line or on a direction-preserving cycle or there is a direction-preserving path $\pi = \langle h = h_0, h_1, \ldots, h_p = k\rangle$ from h to k such that k is an endpoint of a line or on a direction-preserving cycle. Now we consider two cases. In case I we suppose that such a π does not exist and in case II we suppose that such a π exists.

Case I. In case I.1 we suppose that h is an endpoint of a line and in case I.2 we suppose that h is on a direction-preserving cycle.

Case I.1. Suppose that h is an endpoint of an hl-line in $\alpha_{RG}(H; M, C)$. By Lemma 3.2 in H given M and C there is an m-connecting path between h and l, on which there is no arrowhead pointing to h or l. It is easy to observe that h is an ancestor of either (1) a collider node on the path or (2) a node that is an endpoint of a line on the path. Thus we have the two following cases:

(1) If h is an ancestor of a collider node t then $h \in \text{an}(C)$ in H since $t \in C \cup \text{an}(C)$. This by Lemma 3.5(a) implies that there is an m-connecting path given M and C between i and j in H.

(2) If h is an ancestor of a node that is an endpoint of a line on the path then by Lemma 3.5(c) there is an m-connecting path given M and C between i and j in H.

By Algorithm 3.1 both cases imply that $i \sim j$ in $\alpha_{RG}(H; M, C)$ and the ij-edge is endpoint-identical to the m-connecting path. Therefore, $\langle i, h, j\rangle$ is not a ribbon, a contradiction.

Case I.2. Suppose that h is on a direction-preserving cycle in $\alpha_{RG}(H; M, C)$. By Lemma 3.3 in H one of the following holds: (1) $h \in \text{an}(h)$; (2) h or a descendant of h is an endpoint of a line; (3) $h \in \text{an}(C)$. Cases (2) and (3) lead to contradiction as explained in case I.1. Therefore, suppose that h is on a direction-preserving cycle in H. This by Lemma 3.5(c) implies that there is an m-connecting path given M and C between i and j in H, which implies that $\langle i, h, j \rangle$ is not a ribbon. This is a contradiction.

Case II. By Lemma 3.3 in H one of the following holds: (1) $h \in \text{an}(k)$; (2) h or a descendant of h is an endpoint of a line; (3) $h \in \text{an}(C)$. Cases (2) and (3) lead to contradiction as explained in case I.1. Hence it holds that $h \in \text{an}(k)$ in H. Now by using this fact and by the same argument as that of case I (for k instead of h) we obtain a contradiction. □

Notice also that for every ribbonless graph H, it holds that $\alpha_{RG}(H; \varnothing, \varnothing) = H$.

3.2.2 Two necessary properties of α_{RG}

Here we establish the two important properties that α_{RG} (or every generating function) must have. In short, it must be well-defined and it must generate a stable class of graphs. We will show later in this chapter that these two properties also hold for the functions we later define as generating other types of stable mixed graphs.

Well-definition of α_{RG}. The following theorem shows α_{RG} is well-defined in the sense that instead of directly generating an RG we can split the nodes that we marginalise over as well as the nodes that we condition on into two parts, first generate the RG related to the first part, then from the generated RG generate the desired RG by marginalising over and conditioning on the second part:

Theorem 3.1. *For a ribbonless graph $H = (N, F)$ and disjoint subsets C, C_1,*

M, and M_1 of N,

$$\alpha_{RG}(\alpha_{RG}(H; M, C); M_1, C_1) = \alpha_{RG}(H; M \cup M_1, C \cup C_1).$$

Proof. ($\Rightarrow$) If there is an edge between i and j in $\alpha_{RG}(\alpha_{RG}(H; M, C); M_1, C_1)$ then, by Lemma 3.2, there is an m-connecting path $\pi = \langle i = i_0, i_1, \ldots, i_{n-1}, i_n = j \rangle$ between i and j given M_1 and C_1 in $\alpha_{RG}(H; M, C)$ that is endpoint-identical to the edge.

For the V-configuration $\langle i, i_1, i_2 \rangle$ on π, again by Lemma 3.2, given M and C there are m-connecting paths π_1 between i and i_1 and π_2 between i_1 and i_2 in H. These paths can be considered m-connecting given $M \cup M_1$ and $C \cup C_1$ and are endpoint-identical to the edges. This implies that if i_1 is collider (or non-collider) on π in $\alpha_{RG}(H; M, C)$ then on the concatenation of π_1 and π_2 in H it remains collider (or non-collider), or there is an arrowhead pointing to it (or no arrowhead pointing to it) from a joint node on π_1 and π_2. We know that if i_1 is non-collider then it is in M_1 and if it is collider then it is in $C_1 \cup \text{an}(C_1)$ in $\alpha_{RG}(H; M, C)$. Nodes in $M_1 \cup C_1$ in $\alpha_{RG}(H; M, C)$ are in $M_1 \cup C_1$ in H. If $i_1 \in \text{an}(C_1)$ in $\alpha_{RG}(H; M, C)$ then, by Lemma 3.3, one of the following holds in H: 1) it is in $\text{an}(C_1)$; 2) it is in $\text{an}(C)$; 3) it is an endpoint of line or an ancestor of a node that is an endpoint of a line. Therefore, by Lemma 3.5, there is an m-connecting path between i and i_2 given $M \cup M_1$ and $C \cup C_1$ in H, which is endpoint-identical to the V-configuration. By induction along π there is an m-connecting path between i and j given $M \cup M_1$ and $C \cup C_1$ in H, which is endpoint-identical to the ij-edge. Therefore, by Algorithm 3.1 there is the same type of ij-edge in $\alpha_{RG}(H; M \cup M_1, C \cup C_1)$.

($\Leftarrow$) If there is an edge between i and j in $\alpha_{RG}(H; M \cup M_1, C \cup C_1)$ then, by Lemma 3.2, there is an m-connecting path π given $M \cup M_1$ and $C \cup C_1$ in H that is endpoint-identical to the ij-edge. All inner nodes of π are either in $M \cup C \cup \text{an}(C)$ or $M_1 \cup C_1 \cup \text{an}(C_1)$. Therefore, π can be partitioned into m-connecting subpaths

given M and C and single nodes, where the endpoints of subpaths and single nodes are in $M_1 \cup C_1 \cup \text{an}(C_1)$. Therefore, by Algorithm 3.1, in $\alpha_{RG}(H; M, C)$ the endpoints of each of the discussed subpaths of π are connected by an edge that is endpoint-identical to the subpath.

In addition, for each collider V-configuration $\langle l, k, h \rangle$, where $k \in \text{an}(C_1)$ in H and by Lemma 3.4, one of the following holds: 1) $k \in \text{an}(C)$ in H; 2) $k \in \text{an}(C_1)$ in $\alpha_{RG}(H; M, C)$. Case (1) implies that there is an endpoint-identical lh-edge to the V-configuration in $\alpha_{RG}(H; M, C)$, which can be used instead of $\langle l, k, h \rangle$ to generate an m-connecting path.

Hence in $\alpha_{RG}(H; M, C)$ there is an m-connecting path given M_1 and C_1 between i and j, which is endpoint-identical to π. Therefore, again by Algorithm 3.1, there is the same type of ij-edge in $\alpha_{RG}(\alpha_{RG}(H; M, C); M_1, C_1)$. □

Stability of graphs generated by α_{RG}. Here we prove the second important property that α_{RG} must have. This property is the core idea in defining RGs and in general stable mixed graphs. The modification applied by the function should generate a graph that induces the marginal and conditional independence model.

Theorem 3.2. *For a ribbonless graph $H = (N, F)$ and disjoint subsets A, B, C, C_1, and M of N,*

$$A \perp_m B \,|\, C \cup C_1 \text{ in } H \iff A \perp_m B \,|\, C_1 \text{ in } \alpha_{RG}(H; M, C).$$

Proof. To show $A \perp_m B \,|\, C \cup C_1$ in H if and only if $A \perp_m B \,|\, C_1$ in $\alpha_{RG}(H; M, C)$, it is enough to show that, between nodes i and j outside $C \cup C_1 \cup M$, there is an m-connecting path given $C \cup C_1$ in H if and only if there is an m-connecting path given C_1 in $\alpha_{RG}(H; M, C)$.

($\Rightarrow$) Suppose that between i and j there is an m-connecting path given $C \cup C_1$ in H. This path can be partitioned into m-connecting subpaths given C and single nodes, where the endpoints of subpaths and single nodes are colliders in

$C_1 \cup \text{an}(C_1)$. In addition, for each collider V-configuration $\langle l, k, h\rangle$, where $k \in \text{an}(C_1)$ in H and by Lemma 3.4, one of the following holds: 1) $k \in \text{an}(C)$ in H; 2) $k \in \text{an}(C_1)$ in $\alpha_{RG}(H; \varnothing, C)$. Case (1) implies that there is an endpoint-identical lh-edge to the V-configuration in $\alpha_{RG}(H; \varnothing, C)$, which can be used instead of $\langle l, k, h\rangle$ to establish an m-connecting path.

Hence by Algorithm 3.1, in $\alpha_{RG}(H; \varnothing, C)$ there is an m-connecting path given C_1 between i and j. The inner non-collider nodes on this path are either in M or in $N \setminus (M \cup C \cup C_1)$. On this path there are subpaths whose inner nodes are non-collider and in M, i.e. m-connecting subpaths given M and $\varnothing$. By Theorem 3.1 after marginalisation over M, $\alpha_{RG}(H; M, C)$ is obtained, in which, by Algorithm 3.1, the endpoints of each of such subpaths are connected by an edge that is endpoint-identical to the subpath. In addition, if a collider node is in $\text{an}(C_1)$ in $\alpha_{RG}(H; \varnothing, C)$, then by Lemma 3.4 (since the conditioning set is empty and case (2) of the lemma does not hold), the collider node remains in $\text{an}(C_1)$ in $\alpha_{RG}(H; M, C)$. Therefore, between i and j there is an m-connecting path given C_1 in $\alpha_{RG}(H; M, C)$.

($\Leftarrow$) Suppose that between i and j there is an m-connecting path π given C_1 in $\alpha_{RG}(H; M, C)$. By Lemma 3.2, for each edge of π, there is an endpoint-identical m-connecting path given M and C in H, which is obviously an m-connecting path given $C \cup C_1$. On the concatenation of these paths and for the endpoints of the paths we have the two following cases: 1) When the endpoints are non-collider or there is no arrowhead pointing to them on the concatenation, they are non-collider on π and therefore outside $M \cup C \cup C_1$; 2) When the endpoints are collider or there is an arrowhead pointing to them, they are collider on π and therefore in $C_1 \cup \text{an}(C_1)$. Therefore, by Lemma 3.5, there is an m-connecting path given $C \cup C_1$ in H. □

Corollary 3.1. *For a ribbonless graph $H = (N, F)$ and disjoint subsets M and*

C of N,

$$\alpha(\mathcal{J}_m(H); M, C) = \mathcal{J}_m(\alpha_{RG}(H; M, C)).$$

Corollary 3.2. *The class of RGs is stable.*

The following result has been known and implicitly discussed in the literature, e.g., see Cox and Wermuth [1996]:

Corollary 3.3. *The classes of UGs and BGs are stable.*

Proof. The result follows from the fact that, from UGs and BGs, Algorithm 3.2 generates UGs and BGs respectively. □

Example. Figure 3.4 illustrates a DAG as well as two subsets M and C of its node set. Figure 3.5(a) illustrates the generated ribbonless graph H using Algorithm 3.2 as well as two subsets M_1 and C_1 of its node set, and Figure 3.5(b) illustrates the RG generated by the algorithm from H.

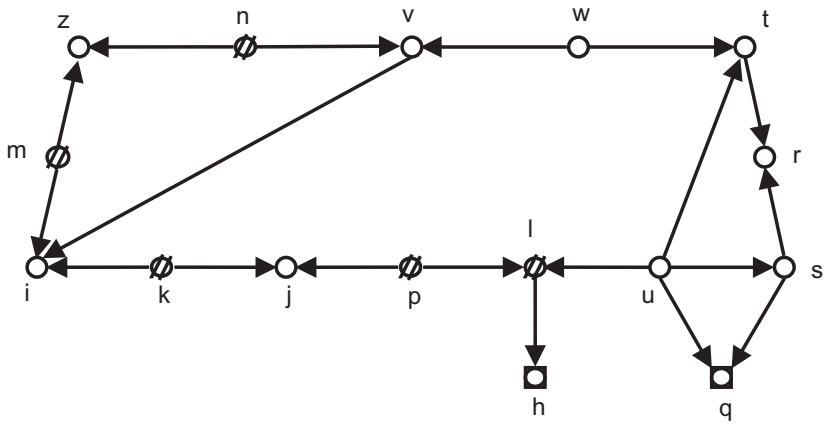
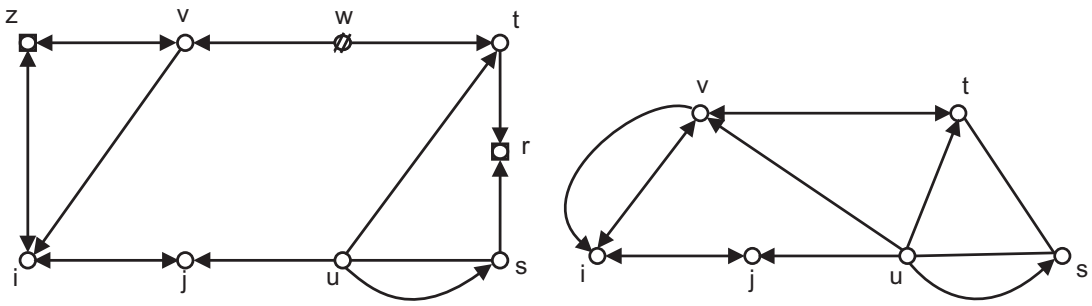


Figure 3.4: A directed acyclic graph G with sixteen nodes, $\not\emptyset \in M$ and $\boxed{\bigcirc} \in C$.

For example, consider the graph $\alpha_{RG}(H, M_1, C_1)$ in Figure 3.5(b), and let $A = \{j\}$, $B = \{s\}$ and $C = \{i, u\}$. It is seen that $v \in \text{an}(C)$. We have that A is not m-separated from B given C since $\langle j, i, v, t, s\rangle$ is an m-connecting



(a) (b)

Figure 3.5: (a) The generated ribbonless graph $H = \alpha_{RG}(G, M, C)$ from the directed acyclic graph G in Figure 3.4, $\not\emptyset \in M_1$ and $\boxed{\bigcirc} \in C_1$. (b) The generated ribbonless graph $\alpha_{RG}(H, M_1, C_1)$ from H.

path between A and B given C. By Theorem 3.2 we conclude that A is not m-separated from B given $C \cup C_1$. The same conclusion can be made by observing m-connecting path $\langle j, i, v, w, t, r, s \rangle$ in H.

3.3 Summary graphs

3.3.1 Generating summary graphs

A local algorithm to generate SGs. We now present a local Algorithm (after determining the ancestor set) to generate an SG from an SG. This algorithm is analogous to the formulations described in matrix form in Wermuth [2011]. To introduce a global algorithm, it is enough to use the three steps of Algorithm 3.1 instead of the first step of the following algorithm. One can also generate SGs from DAGs by using the following algorithm.

Algorithm 3.3. $\alpha_{SG}(H; M, C)$*: (Generating an SG from a summary graph* H*) Start from* H*. Label the nodes in* $\mathrm{an}(C)$*.*

1. *Apply Algorithm 3.2.*
2. *Remove all edges (arrows or arcs) with an arrowhead pointing to a node in* $\mathrm{an}(C)$*, and replace these by the edge with the arrowhead removed (line or*

arrow) if the edge does not already exist.

Continually apply step 1 until it is not possible to apply the given step further before moving to the second step.

Figure 3.6 illustrates how to apply Algorithm 3.3 step by step to a DAG. Notice that as it is stated in the description of the algorithm the order of applying the steps does matter here.

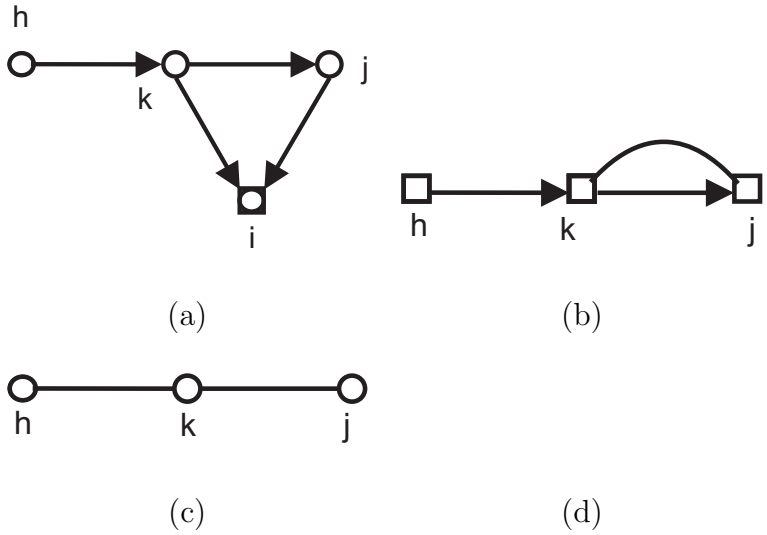


Figure 3.6: (a) A directed acyclic graph G, $\boxed{\bigcirc} \in C$. (b) The generated graph after applying step 1, $\square \in \mathrm{an}(C)$. (c) The generated SG after applying step 2.

The map α_{SG} and its basic properties. Before defining α_{SG} we prove that Algorithm 3.3 generates SGs. Other basic properties of this algorithm and its corresponding function are analogous to the basic properties of RGs.

Proposition 3.3. *Graphs generated by Algorithm 3.3 are SGs.*

Proof. Firstly, the graph generated by the algorithm has only the three desired types of edges and no multiple edge of the same type; therefore, it generates a mixed graph.

Let H be the input summary graph and $H_0 = \alpha_{RG}(H; M, C)$ be the generated

graph after applying step 1 of Algorithm 3.3. Here in part I we prove that there is no armor in the graph, i.e. there is no arrowhead pointing to lines, and in part II we prove that there is no direction-preserving cycle in the generated graph.

Part I. There is no arrowhead pointing to lines in the generated graph: If, for contradiction, there is an arrowhead pointing to i (on an ik-edge) and i is an endpoint of an ij-line in the generated graph then we have the following cases: 1) The ij-line is an arrow from i to j in H_0; 2) the ij-line is an arrow from j to i in H_0; 3) the ij-line is an arc in H_0; 4) the ik-edge is an arrow from k to i in the generated graph and an arc in H_0; 5) the ki- and ij-edges are the same in H_0.

1) We have that $j \in \text{an}(C)$ in H. Hence by Lemma 3.3 we have one of the two following cases in H: a) i or a descendant of i is an endpoint of a line, which, since H is a summary graph, is a contradiction; b) $i \in \text{an}(C)$, which by the algorithm implies that in the generated graph there is no arrowhead pointing to i on the ik-edge, again a contradiction.

2,3) We have that $i \in \text{an}(C)$ in H, which by the algorithm implies that in the generated graph there is no arrowhead pointing to i on the ik-edge, a contradiction.

4,5) We have that $\langle k, i, j \rangle$ is still an armor in H_0. By Lemma 3.2 there is an arrowhead pointing to i in H, hence the line has been generated by the algorithm. Again by Lemma 3.2 we observe that one of the following holds: a) i or a descendant of i is an endpoint of a line, which, since H is a summary graph, is a contradiction; b) $i \in \text{an}(C)$, which by the algorithm implies that in the generated graph there is no arrowhead pointing to i on the ik-edge, again a contradiction.

Part II. There is also no direction-preserving cycles in the generated graph: If, for contradiction, there is a direction-preserving cycle in the generated graph then at least one arrow, say arrow from k to l, should be generated by the algorithm

since there is no direction-preserving cycle in the original SG. This arrow can be generated either by step 1, or by step 2 as an arc replaced by an arrow. If the arrow from k to l is generated by step 2 then we have that $k \in \text{an}(C)$ in H. Therefore, there are no arrowheads pointing to k in the generated graph, which means that k cannot be on a direction-preserving cycle, a contradiction.

Therefore, we can assume that the direction-preserving cycle exists in H_0. Since there are no armors in H and H does not contain a direction-preserving cycle, Lemma 3.3 implies that a node (and therefore all nodes) of the cycle are in $\text{an}(C)$. Hence the arrows turn into lines in the generated graph, a contradiction. □

We have that $\mathcal{SG} \subset \mathcal{RG}$. We consider $\alpha_{SG}(G; M, C)$ to be a function from the sets of SGs and two subsets of their node set to the set of SGs.

The map $\alpha_{RG.SG}$ and its properties. Notice that step 1 of Algorithm 3.3 generates an RG. Hence step 2 of the algorithm generates an SG from an RG and some extra nodes that are conditioned on. We denote this step by $\alpha_{RG.SG}$. This shows that for generating RGs from SGs, $\text{an}(C)$ is needed.

Proposition 3.4. *Let $H = (N, E)$ be a ribbonless graph, and M and C be subsets of N. It holds that $\alpha_{SG}(H; M, C) = \alpha_{RG.SG}(\alpha_{RG}(H; M, C); \text{an}(C))$.*

Computational complexity of the algorithm. We previously showed that the computational complexity of the first step of Algorithm 3.3 is $O(\binom{n^2}{2} n^2(|M| + |C \cup \text{an}(C)|))$. With the same argument as the one for finding $C \cup \text{an}(C)$, the second step can be performed in $|S|^2 f$ steps, where S is the set of C and its ancestors in the generated graph before removing the nodes, and f is the number of edges of the generated graph. A simple conclusion is that Algorithm 3.3 is computationally polynomial.

3.3.2 Two necessary properties of α_{SG}

Here we prove two important results that have been introduced for graphs generated by α_{RG} for graphs generated by α_{SG}.

Well-definition of α_{SG}. This property is analogous to the well-definition of α_{RG} as defined in the previous section. This was discussed implicitly in Wermuth [2011].

Theorem 3.3. *For a summary graph $H = (N, F)$ and disjoint subsets C, C_1, M, and M_1 of N,*

$$\alpha_{SG}(\alpha_{SG}(H; M, C); M_1, C_1) = \alpha_{SG}(H; M \cup M_1, C \cup C_1).$$

Proof. It is enough to show that there is an edge between i and j in $\alpha_{SG}(H; M \cup M_1, C \cup C_1)$ if and only if there is the same type of edge in $\alpha_{SG}(\alpha_{SG}(H; M, C); M_1, C_1)$. For this purpose for summary graphs it is enough to prove that there is an edge between i and j in $\alpha_{SG}(H; M \cup M_1, C \cup C_1)$ if and only if there is an edge between i and j in $\alpha_{SG}(\alpha_{SG}(H; M, C); M_1, C_1)$, and on the ij-edge there is an arrowhead pointing to j in $\alpha_{SG}(H; M \cup M_1, C \cup C_1)$ if and only if there is an arrowhead pointing to j in $\alpha_{SG}(\alpha_{SG}(H; M, C); M_1, C_1)$. This is because the only multiple edge in SGs consists of an arrow and an arc. If there is an arrowhead pointing to i and there is no arrowhead pointing to i on the ij-edge then it is obvious that the ij-edge is a multiple edge, and by the last statement this holds in both mentioned graphs.

($\Rightarrow$) Suppose that there is an ij-edge with arrowhead pointing to j in $\alpha_{SG}(H; M \cup M_1, C \cup C_1)$. We have that $j \notin \text{an}(C \cup C_1)$ in H and in $\alpha_{RG}(H; M \cup M_1, C \cup C_1)$ there is an ij-edge with arrowhead pointing to j. By Lemma 3.2 there is an m-connecting path between i and j given $M \cup M_1$ and $C \cup C_1$ in H with arrowhead pointing to j. By what we showed before in the proof of Theorem 3.1 in $\alpha_{RG}(H; M, C)$ there is an m-connecting path π between i and j given M_1 and C_1

with arrowhead pointing to j.

Now notice that by step 2 of Algorithm 3.3 non-collider nodes remain non-collider. In addition, if a collider V-configuration $\langle h, k, l\rangle$ on π turns into non-collider then $k \in \text{an}(C)$ and therefore, by step 1 of the algorithm there is an endpoint-identical hl-edge that can be used instead of the V-configuration to generate an m-connecting path. Moreover, if the collider node k is in $\text{an}(C_1)$, and on the direction-preserving path an arrow turns into a line then $k \in \text{an}(C)$ in H and again there is an hl-edge to be used instead of the V-configuration for establishing an m-connecting path. Therefore, there is an m-connecting path between i and j given M_1 and C_1 in $\alpha_{SG}(H; M, C)$. We also have that since $j \notin \text{an}(C)$ in H, there is an arrowhead pointing j on the path.

Now by Algorithm 3.1 in $\alpha_{RG}(\alpha_{SG}(H; M, C); M_1, C_1)$ there is an ij-edge with arrowhead pointing j. We also show that in $\alpha_{SG}(H; M, C)$, $j \notin \text{an}(C_1)$: This is because if, for contradiction, $j \in \text{an}(C_1)$ in $\alpha_{SG}(H; M, C)$ then, $j \in \text{an}(C_1)$ in $\alpha_{RG}(H; M, C)$ or $j \in \text{an}(C)$ in H. This by Lemma 3.3 and the fact that H is a summary graph implies that $j \in \text{an}(C \cup C_1)$ in H, a contradiction.

Therefore, since in $\alpha_{SG}(H; M, C)$, $j \notin \text{an}(C_1)$, there is an arrowhead pointing j on the ij-edge in $\alpha_{SG}(\alpha_{SG}(H; M, C); M_1, C_1)$.

($\Leftarrow$) Conversely, suppose that there is an ij-edge with arrowhead pointing to j in $\alpha_{SG}(\alpha_{SG}(H; M, C); M_1, C_1)$. This implies that $j \notin \text{an}(C_1)$ in $\alpha_{SG}(H; M, C)$. In $\alpha_{RG}(\alpha_{SG}(H; M, C); M_1, C_1)$ there is also an ij-edge with arrowhead pointing to j. By Lemma 3.2, in $\alpha_{SG}(H; M, C)$ there is an m-connecting path π given M_1 and C_1 between i an j with arrowhead pointing to j on the path. This implies that $j \notin \text{an}(C)$ in H.

By step 2 of the algorithm, collider nodes on π in $\alpha_{SG}(H; M, C)$ are colliders in $\alpha_{RG}(H; M, C)$. In addition, if a non-collider V-configuration $\langle h, k, l\rangle$ on π is

collider in $\alpha_{RG}(H; M, C)$ then $k \in \text{an}(C)$ in H and, therefore, by step 1 of the algorithm there is an endpoint-identical hl-edge that can be used instead of the V-configuration to generate an m-connecting path in $\alpha_{RG}(H; M, C)$. Moreover, if the collider node k on π is in $\text{an}(C_1)$, and on the direction-preserving path an arrow is an arc in $\alpha_{RG}(H; M, C)$ then the collider node is in $\text{an}(C)$ in H and again there is an hl-edge to be used instead of the V-configuration for establishing an m-connecting path. Therefore, there is an m-connecting path between i and j given M_1 and C_1 in $\alpha_{RG}(H; M, C)$ with arrowhead pointing to j on the path. By what we showed before in the proof of Theorem 3.1 in H there is an m-connecting path between i and j given $M \cup M_1$ and $C \cup C_1$ with arrowhead pointing to j on the path.

Algorithm 3.1 implies that in $\alpha_{RG}(H; M \cup M_1, C \cup C_1)$ there is an ij-edge with arrowhead pointing to j. We showed before that $j \notin \text{an}(C)$ in H. In addition, in H, $j \notin \text{an}(C_1)$: Suppose, for contradiction, that $j \in \text{an}(C_1)$ in H. This direction-preserving path is also direction-preserving in $\alpha_{RG}(H; M, C)$ unless possibly a node t on the path is in M or in C. In the former case one can skip t and obtain a direction-preserving path. In the latter case $j \in \text{an}(C)$ in H, which is not permissible. Therefore, in $\alpha_{RG}(H; M, C)$, $j \in \text{an}(C_1)$. Hence, since $j \notin \text{an}(C)$ in H, $j \in \text{an}(C_1)$ in $\alpha_{SG}(H; M, C)$, a contradiction.

Therefore, $j \notin \text{an}(C \cup C_1)$ in H, and the ij-edge has arrowhead pointing to j in $\alpha_{SG}(H; M \cup M_1, C \cup C_1)$. □

Stability of graphs generated by α_{SG}. We prove that analogous to RGs, graphs generated by α_{SG} induce the marginal and conditional independence model. This result can be implied from what was discussed in Wermuth [2011].

Theorem 3.4. *For a summary graph $H = (N, F)$ and disjoint subsets A, B, C,*

C_1, and M of N,

$$A \perp_m B \,|\, C_1 \text{ in } \alpha_{SG}(H; M, C) \iff A \perp_m B \,|\, C \cup C_1 \text{ in } H.$$

Proof. By what we proved in Theorem 3.2, it is enough to show between i and j there is an m-connecting path given C_1 in $\alpha_{RG}(H; M, C)$ if and only if there is an m-connecting path given C_1 in $\alpha_{SG}(H; M, C)$.

($\Rightarrow$) Consider an m-connecting path given C_1 in $\alpha_{RG}(H; M, C)$ between i and j. By step 2 of the algorithm non-collider nodes remain non-collider. In addition, if a collider V-configuration $\langle h, k, l\rangle$ on π turns into non-collider then $k \in \text{an}(C)$ and, therefore, by step 1 of the algorithm there is an endpoint-identical hl-edge generated that can be used instead of the V-configuration to generate an m-connecting path. Moreover, if the collider node k is in $\text{an}(C_1)$, and on the direction-preserving path an arrow turns into a line then $k \in \text{an}(C)$ in H and again there is an hl-edge to be used instead of the V-configuration for establishing an m-connecting path. Therefore, there is an m-connecting path between i and j given C_1 in $\alpha_{SG}(H; M, C)$.

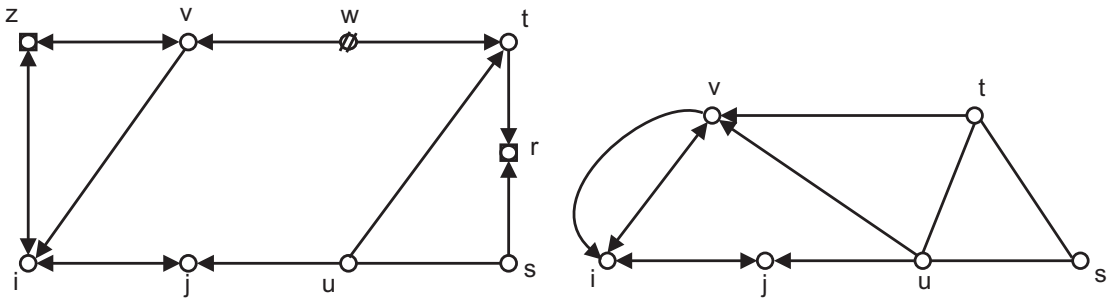
($\Leftarrow$) Consider an m-connecting path π given C_1 in $\alpha_{SG}(H; M, C)$ between i and j. By step 2 of the algorithm, collider nodes on π in $\alpha_{SG}(H; M, C)$ are colliders in $\alpha_{RG}(H; M, C)$. In addition, if a non-collider V-configuration $\langle h, k, l\rangle$ on π is collider in $\alpha_{RG}(H; M, C)$ then $k \in \text{an}(C)$ in H and, therefore, by step 1 of the algorithm there is an endpoint-identical hl-edge that can be used instead of the V-configuration to generate an m-connecting path in $\alpha_{RG}(H; M, C)$. Moreover, if the collider node k on π is in $\text{an}(C_1)$, and on the direction-preserving path an arrow is an arc in $\alpha_{RG}(H; M, C)$ then $k \in \text{an}(C)$ in H and again there is an hl-edge to be used instead of the V-configuration for establishing an m-connecting path. Therefore, there is an m-connecting path between i and j given C_1 in $\alpha_{RG}(H; M, C)$. □

Corollary 3.4. *For a summary graph $H = (N, F)$ and disjoint subsets M and C of N,*

$$\alpha(\mathcal{J}_m(H); M, C) = \mathcal{J}_m(\alpha_{SG}(H; M, C)).$$

Corollary 3.5. *The class of SGs is stable.*

Example. Figure 3.7(a) illustrates the generated SG from the DAG in Figure 3.4 using Algorithm 3.3 as well as the two subsets M_1 and C_1 of its node set. Figure 3.7(b) illustrates the SG generated by the algorithm from the SG in part (a).



(a) (b)

Figure 3.7: (a) The generated SG from the DAG in Figure 3.4, $\not\emptyset \in M_1$ and $\boxed{\bigcirc} \in C_1$. (b) The generated SG from the SG in (a).

3.4 Ancestral graphs

3.4.1 Generating ancestral graphs

A global algorithm to generate AGs. The following algorithm, explained in Richardson and Spirtes [2002], generates an AG from another AG. In fact this algorithm generates a MAG from a given AG. To introduce the algorithm we need the definition of an inducing path. A path between i and j is called *inducing* with

respect to C and M if all its collider nodes are in C or in $\text{an}(\{i,j\} \cup C)$ and all its non-collider nodes are in M.

Algorithm 3.4. $\alpha_{MAG}(H; M, C)$*: (Generating an AG from an ancestral graph* H*) Start from* H*.*

1. *Generate an edge between each node* i *and* j *if* $i \not\sim j$ *and if there is an inducing path between* i *and* j *with respect to* C *and* M *in* H*.*
2. *Let an edge between* i *and* j *be a line if* $i \in \text{ant}(\{j\} \cup C)$ *and* $j \in \text{ant}(\{i\} \cup C)$*, be an arrow from* j *to* i *if* $j \in \text{ant}(\{i\} \cup C)$ *and* $i \notin \text{ant}(\{j\} \cup C)$*, and be an arc if* $i \notin \text{ant}(\{j\} \cup C)$ *and* $j \notin \text{ant}(\{i\} \cup C)$*.*

Continually apply step 1 until it is not possible to apply the given step further before moving to the second step. Then remove all nodes in $M \cup C$*.*

Figure 3.8 illustrates how to apply Algorithm 3.4 step by step to a DAG.

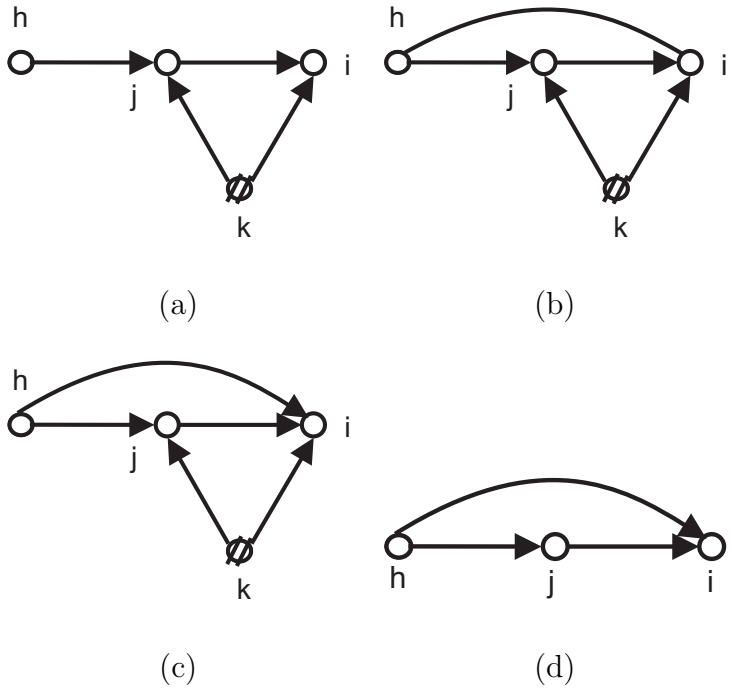


Figure 3.8: (a) A directed acyclic graph G, $\not\emptyset \in M$. (b) The generated graph after applying step 1 (for inducing path (h, j, k, i)). (c) The generated graph after applying step 2 (since $h \in \text{an}(i)$). (d) The generated AG from G.

A local algorithm to generate AGs. Here we show that there is a way of generating AGs by looking only for V-configurations. The following algorithm gives a local way of generating AGs from AGs after determining the ancestor set:

Algorithm 3.5. $\alpha_{AG}(H; M, C)$*: (Generating an AG from an ancestral graph* H*) Start from* H*.*

1. *Apply Algorithm 3.3.*
2. *Generate an endpoint-identical edge to a collider V-configuration whose inner node is an ancestor of whose endpoint, i.e., generate respectively an arrow from* j *to* i *or an arc between* i *and* j *for V-configuration* $j \tuh k \huh i$ *or V-configuration* $j \huh k \huh i$ *when* $k \in \mathrm{an}(i)$ *if the arrow or the arc does not already exist.*
3. *Remove the arc between* j *and* i *in the case that* $j \in \mathrm{an}(i)$*, and replace it by an arrow from* j *to* i *if the arrow does not already exist.*

Continually apply each step until it is not possible to apply the given step further before moving to the next step.

Figure 3.9 illustrates how to apply Algorithm 3.5 step by step on a DAG. The following lemma interprets this local algorithm globally. A *non-ancestral inducing path* is an inducing path $\langle i = i_0, i_1, \ldots, i_m = j \rangle$ that can be partitioned into two subpaths $\langle i, i_1, \ldots, i_l \rangle$ and $\langle i_{l+1}, \ldots, i_m \rangle$ (where one subpath can be empty) such that i_n, $1 \leq n \leq l$, that are collider are in $C \cup \mathrm{an}(C \cup \{i\})$ and i_p, $l+1 \leq p \leq m-1$, that are collider are in $C \cup \mathrm{an}(C \cup \{j\})$.

Lemma 3.6. *Let* H *be a summary (or ancestral) graph and* M *and* C *two disjoint subsets of its node set. There is an edge between* i *and* j *with no arrowhead at* i *in* $\alpha_{AG}(H; M, C)$ *if and only if there is a non-ancestral inducing path with respect to* C *and* M *between* i *and* j *and* $i \in \mathrm{ant}(\{j\} \cup C)$ *in* H*.*

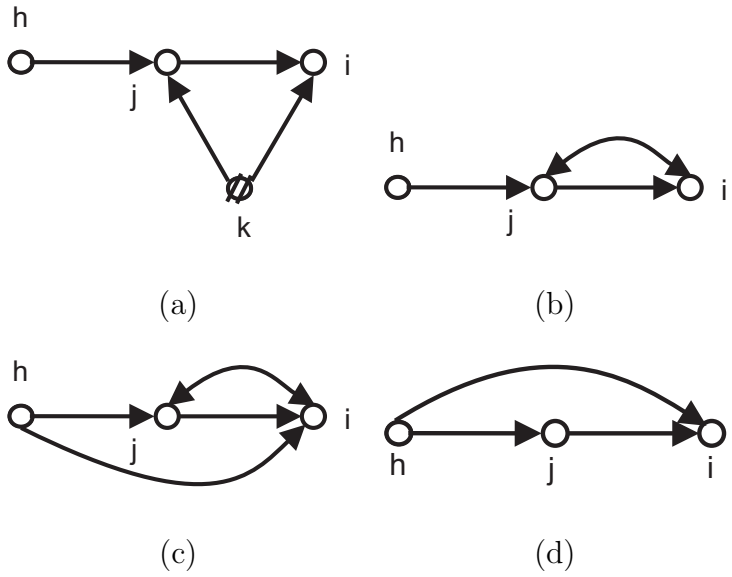


Figure 3.9: (a) A directed acyclic graph G, $\not\emptyset \in M$. (b) The generated summary graph after applying step 1. (c) The generated graph after applying step 2 for V-configuration $\langle h, j, i \rangle$. (d) The generated AG from G after applying step 3.

Proof. ($\Rightarrow$) Suppose that there is an edge between i and j with no arrowhead at i in $\alpha_{AG}(H; M, C)$. Step 3 of Algorithm 3.5 only changes the type of edge. Therefore, there is an ij-edge before applying step 3 of Algorithm 3.5. In addition, before applying step 3 of Algorithm 3.5 there is no arrowhead at i on the edge or i has been an arc turning into arrow from i to j, which implies that $i \in \text{an}(j)$. Both cases imply that $i \in \text{ant}(j)$.

Now let H_0 be the generated graph after applying step 1 of Algorithm 3.5. In addition, let $\langle H_0, H_1, \ldots, H_m \rangle$ be intermediate graphs that have each been generated by adding one edge to the previous graph by step 2 of Algorithm 3.5. We prove by reverse induction along this sequence that in each H_n, $0 \leq n \leq m$ there is a non-ancestral primitive inducing path between i and j and $i \in \text{ant}(j)$. For H_m the result trivially holds. Now suppose that there is a non-ancestral primitive inducing path between i and j and $i \in \text{ant}(j)$ in H_n. We prove that there is such a path in H_{n-1} and $i \in \text{ant}(j)$. Suppose that a qs-edge is generated at

this step. Hence in H_{n-1} there is an endpoint-identical collider V-configuration $\langle q, r, s\rangle$, where $r \in \text{an}(s)$. If the qs-edge is on the non-ancestral primitive inducing path then $s \in \text{an}(\{i, j\})$. Therefore, $r \in \text{an}(\{i, j\})$ and by replacing the qs-edge by $\langle q, r, s\rangle$ we obtain a primitive inducing path. The path remains non-ancestral since r is between q and s on the path. If the qs-edge is an arrow from q to s on the direction-preserving path from a collider node on the path to i or j then one can replace the qs-arrow by the arrow from q to r together with the direction-preserving path from r to s. Therefore, there is a non-ancestral primitive inducing path or edge between i and j in H_{n-1}, and by reverse induction, in H_0. In addition, by the same argument an arrow or an arc on the path that turns into an anterior path from i to j in the anterior graph can be replaced by the qr-edge together with the direction-preserving path from r to s to imply that $i \in \text{ant}(j)$ in H_0.

Now we proceed to applying step 1 of Algorithm 3.5. Within this step we have step 2 of Algorithm 3.3. The only way that this step can generate a primitive inducing path in H_0 is to replace a uv-arc by an arrow from u to v to generate a direction-preserving path from a collider node y on $\langle w, y, z\rangle$ on the path to i. This implies that $u \in \text{an}(C)$ in H. This by Lemma 3.3 implies that $y \in \text{an}(C)$ in H. Therefore, by step 1 of Algorithm 3.3 there is an endpoint-identical wz-edge that can be used instead of $\langle w, y, z\rangle$ to establish a non-ancestral primitive-inducing path. In addition, this implies that if an arrow from v to u or a uv-arc is on a path that turns into an anterior path from i to j in the anterior graph and if it turns into a line or an arrow from u to v at this step then $u \in \text{an}(C)$ in H and, therefore, $i \in \text{ant}(C)$ in H.

Now for each edge of the primitive inducing path there is an endpoint-identical m-connecting path given M and C in H, which is obviously inducing with respect to C and M. The endpoints of these m-connecting paths are in $C \cup \text{an}(C \cup \{i\})$.

On the concatenation of these m-connecting paths the endpoints are collider or there is an arrowhead pointing to them, which, by Lemma 2.11 (by considering $C' = C \cup \{i\}$ or $C' = C \cup \{j\}$) implies that there is a non-ancestral inducing path between i and j with respect to C and M in H. In addition, we know that $i \in \text{ant}(C)$ in H or $i \in \text{ant}(j)$ after applying step 1 of Algorithm 3.3. These, by Lemma 3.3 and the fact that H is armorless, imply that $i \in \text{ant}(\{j\} \cup C)$ in H.

($\Leftarrow$) Suppose that there is a non-ancestral inducing path π between i and j with respect to C and M and $i \in \text{ant}(\{j\} \cup C)$ in H. The path π can be partitioned into m-connecting subpaths given M and C and collider nodes in $\text{an}(\{i, j\})$. After applying step 1 of Algorithm 3.3 to H, the endpoints of all the subpaths become connected. By Lemma 3.4, after applying this step, collider nodes are either in $\text{an}(C)$ or in $\text{an}(\{i, j\})$. If the inner node of a collide V-configuration is in $\text{an}(C)$ then there is an endpoint-identical edge between its endpoints that can be used instead of the V-configuration. Therefore, after applying step 1 of Algorithm 3.3, there is a non-ancestral primitive inducing path between i and j. In addition, if $i \in \text{ant}(\{j\} \cup C) \setminus \text{an}(\{j\} \cup C)$ then there is no arrowhead at i since H is an SG. Hence there is no arrowhead at i or $i \in \text{an}(\{j\} \cup C)$.

Denote a shortest non-ancestral primitive inducing path between i and j by π_0. Notice that the inner colliders on π_0 are not ancestors of C since otherwise, by step 1 of Algorithm 3.3, there is a shorter path between i and j. Therefore, after applying step 2 of Algorithm 3.3, π_0 remains a non-ancestral primitive inducing path between i and j. In addition, If $i \in \text{an}(C)$ then the arrowheads at i are removed. Hence there is no arrowhead at i or $i \in \text{an}(j)$.

It is easy to observe that, after repeatedly applying step 2 of Algorithm 3.5 to a non-ancestral primitive inducing path, an endpoint-identical edge is generated. In addition, on the generated ij-edge there is no arrowhead at i or, since no edge is replaced by this step, $i \in \text{an}(j)$.

The ij-edge also exists after applying step 3 of Algorithm 3.5. In addition, if $i \in \text{an}(j)$ then the arrowhead at i is removed by this step. Therefore, there is an edge between i and j with no arrowhead at i in $\alpha_{AG}(H; M, C)$. □

Relationship between α_{AG} and α_{MAG}. In spite of Algorithm 3.4, which generates MAGs, Algorithm 3.5 may generate non-maximal AGs. However, by applying Algorithm 2.3, -which generates MAGs from AGs-, to the output of Algorithm 3.5 we get the same MAG as that of Algorithm 3.4. We consider Algorithm 2.3 to be a function and denote it by $\alpha_{AG.MAG}$.

Proposition 3.5. *It holds that $\alpha_{MAG} = \alpha_{AG.MAG} \circ \alpha_{AG}$.*

Proof. For an ancestral graph H and two subsets of its node set M and C, it is enough to show that there is an edge between i and j with no arrowhead at i in $\alpha_{MAG}(H; M, C)$ if and only if there is an edge between i and j with no arrowhead at i in $\alpha_{AG.MAG}(\alpha_{AG}(H; M, C))$.

($\Rightarrow$) Suppose that there is an edge between i and j in $\alpha_{MAG}(H; M, C)$ with no arrowhead at i. By algorithm 3.4 there is an inducing path π with respect to C and M in H, and $i \in \text{ant}(\{j\} \cup C)$. If this path is non-ancestral then Lemma 3.6 implies that there is an ij-edge in $\alpha_{AG}(H; M, C)$. If it is not then partition this path into inducing subpaths with collider nodes in $\text{an}(C)$ (i.e. m-connecting paths given M and C) and collider nodes that are in $\text{an}(\{i, j\}) \setminus \text{an}(C)$. Consider one of the subpaths between k and l. Notice that k and l are not in $\text{an}(C)$. Lemma 3.6 implies that there is a kl-edge in $\alpha_{AG}(H; M, C)$, which has an arrowhead at k unless $k \in \text{an}(l)$ in H. However, if $k \in \text{an}(l)$ in H then, since $k \notin \text{an}(C)$, $k \in \text{an}(l)$ after applying step 1 of Algorithm 3.5. Step 2 implies that there is an endpoint-identical edge between l and the other node adjacent to k on the path. Hence k can be skipped to obtain a shorter path. This implies that there is an ancestral primitive inducing path between i and j in $\alpha_{AG}(H; M, C)$ with no arrowhead at i on the path. Therefore, by Algorithm 2.3, there is an ij-edge with

no arrowhead at i in $\alpha_{AG.MAG}(\alpha_{AG}(H; M, C))$.

($\Leftarrow$) Suppose that there is an edge between i and j with no arrowhead at i in $\alpha_{AG.MAG}(\alpha_{AG}(H; M, C))$. Therefore, there is a primitive inducing path between i and j with no arrowhead at i in $\alpha_{AG}(H; M, C)$. Now by Lemma 3.6, for each kh-edge of the primitive inducing path, there is an inducing path with respect to C and M between k and h in H such that $k \notin \text{ant}(\{h\} \cup C)$. Notice that there are arrowheads pointing to the endpoints of these inducing paths, since none of the steps of Algorithm 3.5 adds arrowheads to edges. In addition, we know that $k \in \text{an}(\{i, j\})$ in $\alpha_{AG}(H; M, C)$. Consider an arrow from k to z on the direction-preserving cycle from k to i or j. By Lemma 3.6, $k \in \text{ant}(\{z\} \cup C)$ in H, but since there are no arrowheads pointing to lines in H and $k \notin \text{ant}(C)$, it holds that $k \in \text{an}(z)$. By induction along the direction-preserving path it holds that $k \in \text{an}(\{i, j\})$ in H.

On the concatenation of these paths, the endpoints of the paths are collider or there is an arrowhead pointing to them. In addition, we showed that these are in $\text{an}(\{i, j\})$ in H. Therefore, by Lemma 3.5 (for $C' = C \cup \{i, j\}$), there is an inducing path between i and j with respect to C and M in H.

In addition, denote the node adjacent to i on the generated inducing path by l_1. By Lemma 3.6, $i \in \text{ant}(C \cup \{l_1\})$. If l_1 is a collider node then $l_1 \in \text{an}(C \cup \{j\})$ and hence $i \in \text{ant}(C \cup \{j\})$. If l_1 is non-collider then $l_1 \in \text{pa}(l_2)$ or $l_1 \in \text{ne}(l_2)$. These imply that $l_1 \in \text{ant}(l_2)$, where l_2 is the node adjacent to l_1 on the inducing path. Therefore, by induction along the inducing path we obtain $i \in \text{ant}(C \cup \{j\})$.

Therefore, by Algorithm 3.4, there is an edge between i and j in $\alpha_{MAG}(H; M, C)$ and there is no arrowhead at i on the ij-edge. □

The map α_{AG} and its basic properties. Here we prove that Algorithm 3.5 generates AGs. Other basic properties of this algorithm and its corresponding

function are analogous to the basic properties of RGs and SGs. Algorithm 3.4 and the following result, just for Algorithm 3.4, were discussed in Richardson and Spirtes [2002].

Proposition 3.6. *Graphs generated by Algorithms 3.4 and 3.5 are AGs.*

Proof. To prove that the graph generated by Algorithm 3.5 is an AG, first notice that graphs generated by step 1 of Algorithm 3.5 are summary graphs. In part I we prove that there are no arrowheads pointing to line in the generated graph. In part II we prove that there is no bow in the graph. In part III we prove that there is no direction-preserving cycle in the generated graph.

Part I. Since steps 2 and 3 do not generate any lines and step 3 does not generate any arrowheads, it is enough to show that the node i on a generated ij-arc by step 2 is not an endpoint of a line. Consider the first iteration of the algorithm, where, for contradiction, such an ij-arc is generated. However, in the previous iteration of the algorithm there is an arrowhead pointing to i on the V-configuration that generates the ij-arc, and i is an endpoint of a line, a contradiction.

Part II. There is no bow in the generated graph since step 3 of the algorithm removes all bows of the graph.

Part III. Step 2 does not generate any direction-preserving cycles by adding an arrow to the graph: Consider the first iteration of the algorithm, where, for contradiction, a direction-preserving cycle is generated. If it is generated by generating an arrow from i to j then we know that there is $i \dashrightarrow k \dashleftarrow\!\dashrightarrow j$, where $k \in \text{an}(j)$. Denote the direction-preserving path from k to j by π_1 and the direction-preserving path from j to i which, together with the generated ij-arrow, establishes a direction-preserving cycle by π_2. It is seen that in the the previous iteration of the algorithm $\langle \pi_2, k, \pi_1 \rangle$ is a direction-preserving cycle, a contradiction.

Step 3 does not generate any direction-preserving cycles by replacing an arc by an arrow: Consider the first iteration of the algorithm, where, for contradiction, a direction-preserving cycle is generated. If it is generated by replacing an ij-arc by an arrow from i to j then we know there is a direction-preserving path π_1 from i to j. Denote the direction-preserving path from j to i which, together with the generated ij-arrow, establishes a direction-preserving cycle by π_2. It is seen that in the the previous iteration of the algorithm $\langle \pi_1, \pi_2 \rangle$ is a direction-preserving cycle, a contradiction.

The fact that the graph generated by Algorithm 3.5 is an AG follows from Proposition 3.5. This result can be seen separately as a consequence of Theorem 4.2 ((i), (ii)) and Definition 4.2.1 in Richardson and Spirtes [2002]. □

As before we consider α_{AG} to be a function from the set of AGs and two subsets of their node set to the set of AGs.

The map $\alpha_{SG.AG}$ and its properties. Notice that step 1 of the algorithm generates an SG. Hence steps 2 and 3 of the algorithm generate an AG from an SG. We denote these two steps by $\alpha_{SG.AG}$.

Proposition 3.7. *It holds that $\alpha_{AG} = \alpha_{SG.AG} \circ \alpha_{SG}$.*

The map $\alpha_{SG.AG}$ and its properties. Notice that if $H \in \mathcal{AG}$ then $\alpha_{SG.AG}(H) = H$. In addition, the following lemma shows that Algorithm 3.5 can be applied to an SG instead of an AG:

Lemma 3.7. *Let H be a summary graph and M and C be two subsets of its node set. It holds that $\alpha_{AG}(\alpha_{SG.AG}(H); M, C) = \alpha_{AG}(H; M, C)$.*

Proof. For a summary graph H and two subsets of its node set M and C, it is enough to show that there is an edge between i and j with no arrowhead at i in $\alpha_{AG}(\alpha_{SG.AG}(H); M, C)$ if and only if there is an edge between i and j with

no arrowhead at i in $\alpha_{AG}(H; M, C)$. By Lemma 3.6, the left hand side and the right hand side are equivalent to the existence of a non-ancestral inducing path with respect to C and M between i and j and $i \in \text{ant}(\{j\} \cup C)$ respectively in $\alpha_{SG.AG}(H)$ and H. Therefore, it is enough to prove that, between i and j and with respect to C and M, there is a non-ancestral inducing path in $\alpha_{SG.AG}(H)$ and $i \in \text{ant}(\{j\} \cup C)$ if and only if there is a non-ancestral inducing path in H and $i \in \text{ant}(\{j\} \cup C)$.

($\Rightarrow$) Suppose that there is a non-ancestral inducing path with respect to C and M between i and j and $i \in \text{ant}(\{j\} \cup C)$ in $\alpha_{SG.AG}(H)$. This path remains non-ancestral inducing before applying step 3 of Algorithm 3.5 unless possibly a kh-edge on the direction-preserving path from a collider node on the path to $\{i, j\} \cup C$ is an arc turning into arrow. In this case $k \in \text{an}(h)$ and the direction-preserving path from k to h can be used instead of the kh-edge. Therefore, there is a non-ancestral inducing path with respect to C and M between i and j before applying step 3 of Algorithm 3.5. This also implies that, before applying step 3 of Algorithm 3.5, $i \in \text{ant}(\{j\} \cup C)$.

Now let $\langle H = H_0, H_1, \ldots, H_m \rangle$ be intermediate graphs that have each been generated by adding one edge in the previous graph by step 2 of Algorithm 3.5. We prove by reverse induction along this sequence that in each H_n, $0 \leq n \leq m$ there is a non-ancestral inducing path between i and j and $i \in \text{ant}(\{j\} \cup C)$. For H_m the result trivially holds. Now suppose that there is a non-ancestral inducing path between i and j and $i \in \text{ant}(\{j\} \cup C)$ in H_n. We prove that there is such a path in H_{n-1} and $i \in \text{ant}(\{j\} \cup C)$. Suppose that a qs-edge is generated at this step. Hence in H_{n-1} there is an endpoint-identical collider V-configuration $\langle q, r, s \rangle$, where $r \in \text{an}(s)$. Now we deal with two cases:

a) If the qs-edge is on the non-ancestral inducing path then we have the following cases: 1) If s is non-collider on the path then, by Lemma 2.6, $s \in \text{an}(\{i, j\})$

or s is an ancestor of a collider node on the path, which implies that $s \in \text{an}(C \cup \{i, j\})$. Therefore, $r \in \text{an}(C \cup \{i, j\})$. 2) If s is collider on the path then $s \in C \cup \text{an}(C \cup \{i, j\})$. Therefore, $r \in \text{an}(C \cup \{i, j\})$. Both cases imply that by replacing the qs-edge by $\langle q, r, s\rangle$ we obtain a non-ancestral inducing path.

b) If the qs-edge is an arrow from q to s on the direction-preserving path from a collider node on the path to $C \cup \{i, j\}$ then one can replace the qs-arrow by the arrow from q to r together with the direction-preserving path from r to s. Therefore, there is a non-ancestral inducing path between i and j in H_{n-1}, and by reverse induction, in H.

In addition, this implies that an arrow on the anterior path from i to j can be replaced by a direction-preserving path, and also since lines are not generated by this step of the algorithm, $i \in \text{ant}(\{j\} \cup C)$ in H.

($\Leftarrow$) Suppose that there is a non-ancestral inducing path with respect to C and M between i and j and $i \in \text{ant}(\{j\} \cup C)$ in H. After applying step 2 of Algorithm 3.5, since only edges may be added, this path still exists in the generated graph and we still have that $i \in \text{ant}(\{j\} \cup C)$.

Now suppose that by step 3 of Algorithm 3.5 a kh-arc on the V-configuration $\langle l, k, h\rangle$ on the inducing path turns into an arrow from k to h. If the V-configuration is non-collider before applying step 3 then the path is still inducing. If the V-configuration is collider before applying step 3 then, since $k \in \text{an}(h)$, by step 2 of the algorithm there is an endpoint-identical lh-edge that can be used instead of the V-configuration to establish an inducing path. In addition, since no lines or arrows are changed by this step, we still have that $i \in \text{ant}(\{j\} \cup C)$. Therefore, there is a non-ancestral inducing path with respect to C and M between i and j in $\alpha_{SG.AG}(H)$ and $i \in \text{ant}(\{j\} \cup C)$. □

Computational complexity of the local algorithm. As what we have shown before, step 1 of Algorithm 3.5 is computationally polynomial with computational complexity $O(n^7)$. The second step of the algorithm, in the worst case scenario, can be performed in $2n^2e^3$ steps, where n is the number of nodes and e the number of edges of the graph. This is because, for each ik-arc, whether i is an ancestor of k can be deduced in $|\text{an}(i)|^2e + |\text{an}(k)|^2e$ steps. If $k \in \text{an}(i)$ then, by going through the edge set, one can check whether an arrow from another node j to k or a jk-arc exists such that j is not respectively in $\text{pa}(i)$ or $\text{sp}(i)$.

By the same method as that of step 2, for each node i, step 3 of the algorithm can be performed in $|\text{an}(i)|^2e^2$ steps. Therefore, in total, step 3 can be performed in n^3e^2 steps in the worst case scenario. Hence the algorithm is computationally polynomial.

3.4.2 Two necessary properties of α_{AG}

Again we discuss the two important properties that we have proven for two other types of stable mixed graphs.

Well-definition of α_{AG}. Well-definition of α_{AG} is analogous to the well-definition of α_{RG} and α_{SG} as defined in the previous sections.

Theorem 3.5. *For an ancestral graph $H = (N, F)$ and disjoint subsets C, C_1, M, and M_1 of N,*

$$\alpha_{AG}(\alpha_{AG}(H; M, C); M_1, C_1) = \alpha_{AG}(H; M \cup M_1, C \cup C_1).$$

Proof. Using Theorem 3.3, Proposition 3.7, and Lemma 3.7, we have the following:

$$\alpha_{AG}(\alpha_{AG}(H; M, C); M_1, C_1) = \alpha_{AG}(\alpha_{SG.AG}(\alpha_{SG}(H; M, C)); M_1, C_1) =$$

$$\alpha_{AG}(\alpha_{SG}(H; M, C); M_1, C_1) = \alpha_{SG.AG}(\alpha_{SG}(\alpha_{SG}(H; M, C); M_1, C_1)) =$$

$$\alpha_{SG.AG}(\alpha_{SG}(H; M \cup M_1, C \cup C_1)) = \alpha_{AG}(H; M \cup M_1, C \cup C_1).$$

□

Stability of graphs generated by α_{AG}. Analogous to RGs and SGs, graphs generated by α_{AG} induce marginal and conditional independence models.

Theorem 3.6. *[Richardson and Spirtes, 2002] For an ancestral graph $H = (N, F)$ and disjoint subsets A, B, C, C_1, and M of N,*

$$A \perp_m B \,|\, C_1 \text{ in } \alpha_{AG}(H; M, C) \iff A \perp_m B \,|\, C \cup C_1 \text{ in } H.$$

Proof. The result follows from Theorem 4.18 of Richardson and Spirtes [2002] and the fact that $\alpha_{AG.MAG}$ does not change the independence model. □

Corollary 3.6. *[Richardson and Spirtes, 2002] For an ancestral graph $H = (N, F)$ and disjoint subsets M and C of N,*

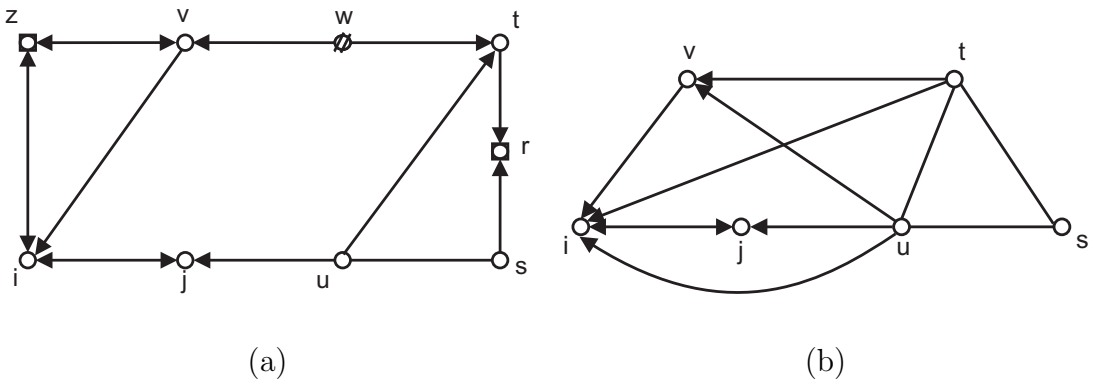
$$\alpha(\mathcal{J}_m(H); M, C) = \mathcal{J}_m(\alpha_{AG}(H; M, C)).$$

Corollary 3.7. *The class of AGs is stable.*

Example. Figure 3.10(a) illustrates the AG generated from the DAG in Figure 3.4. Figure 3.10(b) illustrates the AG generated by the algorithm from the AG in part (a).

3.5 The relationship between different types of stable mixed graphs

Goal of the section. Thus far, we have defined RGs (as a modification of MC graphs), SGs, and AGs, and introduced algorithms to generate each of them from a graph of the same class or a DAG, and some algorithms that act between the classes. Despite the similarities of these definitions and generating algorithms of



(a) (b)

Figure 3.10: (a) The generated AG from the DAG in Figure 3.4, $\not\emptyset \in M_1$ and $\boxed{\bigcirc} \in C_1$. (b) The generated AG from the AG in (a).

these different classes, as well as the parallel theory developed for them, it is of interest to investigate the exact relationship between these types of graphs.

In order to scrutinise the relationship between three types of stable mixed graphs, one has to envisage relationships between their generating algorithms. For this purpose, the main question is how we can or why we cannot find functions between different images $\alpha(G; M, C)$ for a directed acyclic graph G and two subsets of its node set M and C.

Another question that needs be demanded is whether the image of generating functions is big enough to cover all graphs included in the set of the related type of stable mixed graphs. This is equivalent to asking whether generating functions are surjective. This question will be tackled and it will become apparent that generating functions are surjective.

Furthermore, by showing that generating functions are not injective, the relationship between different types of stable mixed graphs will be summarised by the diagram in Figure 3.11, in which one can only move towards the directions that arrows show.

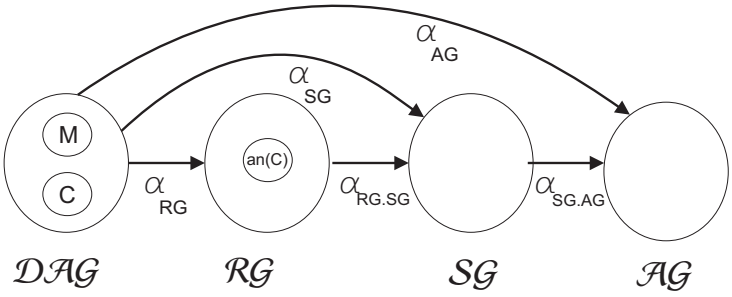


Figure 3.11: The relationship between $\mathcal{DAG}$, $\mathcal{RG}$, $\mathcal{SG}$, and $\mathcal{AG}$.

3.5.1 Corresponding types of stable mixed graphs

Definition of corresponding stable mixed graphs. When one starts from a DAG and generates different types of stable mixed graphs after marginalisation over and conditioning on two specific subsets of the node set of the DAG, the generated graphs must induce the same independence models. This leads us to the definition of corresponding stable mixed graphs. For a directed acyclic graph G and two disjoint subsets of its node set M and C, graphs $\alpha_{RG}(G; M, C)$, $\alpha_{SG}(G; M, C)$, and $\alpha_{AG}(G; M, C)$ are called respectively the *corresponding* RG, SG, and AG.

The two graphs in Figure 3.7 are the corresponding SGs to RGs in Figure 3.5, and the two graphs in Figure 3.10 are the corresponding AGs to RGs in Figure 3.5 and SGs in Figure 3.7.

We observe that the corresponding RGs, SGs, and AGs of a DAG induce the same independence model. This fact, without being formulated in this way, was discussed in all three papers that define these graphs [Koster, 2002, Richardson and Spirtes, 2002, Wermuth, 2011].

Proposition 3.8. *For a directed acyclic graph $G = (V, E)$ and disjoint subsets*

C and M of V,

$$\mathcal{J}_m(\alpha_{RG}(G; M, C)) = \mathcal{J}_m(\alpha_{SG}(G; M, C)) = \mathcal{J}_m(\alpha_{AG}(G; M, C)).$$

Proof. The result follows from Corollaries 3.1, 3.4, and 3.6. □

The relationship between corresponding RGs, SGs, and AGs. We cannot generate the corresponding SG to a given RG by only knowing the RG and not the DAG, or at least its conditioning set.

This is because there is no function f such that $f \circ \alpha_{RG} = \alpha_{SG}$. Let the directed acyclic graph G_1 be $\circ \leftarrow \not\emptyset \rightarrow \circ \rightarrow \boxed{\circ} \leftarrow \not\emptyset$, $\not\emptyset \in M$ and $\boxed{\circ} \in C$. We have $\alpha_{RG}(G_1; M, C) = \circ \leftrightarrow \circ$ and $\alpha_{SG}(G_1; M, C) = \circ \leftarrow \circ$. Therefore, $f(\circ \leftrightarrow \circ) = \circ \leftarrow \circ$. Now let that the graph G_2 be $\circ \leftrightarrow \circ$. We have $\alpha_{RG}(G_2; \varnothing, \varnothing) = \circ \leftrightarrow \circ$ and $\alpha_{SG}(G_2; \varnothing, \varnothing) = \circ \leftrightarrow \circ$. Therefore, $f(\circ \leftrightarrow \circ) = \circ \leftrightarrow \circ$. This result is also true for AGs instead of SGs.

However, it is possible to give an algorithm to generate SGs that induce the same independence model as the given RGs by generating their anterior graphs. We will not explain its details in this thesis. Proposition 3.7, however, shows that we can generate the corresponding AG to a given SG.

3.5.2 Properties of functions generating stable mixed graphs

Non-injectivity of generating functions. Graphs $\circ \rightarrow \circ$ and $\not\emptyset \rightarrow \circ \rightarrow \circ$, $\not\emptyset \in M$, reveal that generating functions are not injective, hence it is not possible to recover the graphs in the diagram in Figure 3.11 in the opposite direction to the DAG.

It is also clear from the generating algorithms that one cannot generate the corresponding RG to a given SG, or the corresponding SG to a given AG as the generating functions are not injective.

Surjectivity of generating functions. However, the generating functions with the set of DAGs as argument are surjective.

Proposition 3.9. *The following functions are surjective:*

a) $\alpha_{RG} : \mathcal{DAG} \to \mathcal{RG}$*;*

b) $\alpha_{SG} : \mathcal{DAG} \to \mathcal{SG}$*;*

c) $\alpha_{AG} : \mathcal{DAG} \to \mathcal{AG}$*.*

Proof. a) Let $H = (N_1, F_1) \in \mathcal{RG}$. We generate a directed graph $G = (V, E)$ from H as follows: We leave arrows that are not on any direction-preserving cycle unchanged. For direction-preserving cycles, instead of one arbitrary arrow from j to i on the cycle we place $j \to \boxed{\circ} \leftarrow \not\circ \to i$, where $\not\circ \in M$ and $\boxed{\circ} \in C$, and leave all other arrows unchanged. Instead of an arc between j and i we place a V-configuration between j and i with inner source node in M. Instead of a line between j and i we place a V-configuration between j and i with inner collider node in C. The graph G is obviously a directed graph. Furthermore, all newly generated nodes have degree 2 and the direction of arrows changes on them, hence these cannot be on any direction-preserving cycle. In addition, if i and j are in N_1 and $i \sim j$ in G then $i \in \mathrm{pa}(j)$ or $j \in \mathrm{pa}(i)$ in H. Therefore, the existence of a direction-preserving cycle in G implies the existence of the same direction-preserving cycle in H. But by the nature of the construction of G we know that direction-preserving cycles in H do not make direction-preserving cycles in G, hence G is acyclic.

We should prove that $\alpha_{RG}(G; M, C) = H$. Let $\alpha_{RG}(G; M, C) = (N_2, F_2)$. Obviously $N_1 = N_2$. Suppose that $i \sim j$ ($j \in \mathrm{pa}(i)$, $j \in \mathrm{sp}(i)$, or $j \in \mathrm{ne}(i)$) in H. Therefore, we have the active alternating path or one of the V-configurations between i and j that by Algorithm 3.2 forms exactly the same type of edge in $\alpha_{RG}(G; M, C)$.

Conversely, suppose that $i \sim j$ ($j \in \mathrm{pa}(i)$, $j \in \mathrm{sp}(i)$, or $j \in \mathrm{ne}(i)$) in $\alpha_{RG}(G; M, C)$. By Lemma 3.2 we know that there is an endpoint-identical m-connecting path given M and C in G. Consider a shortest endpoint-identical m-connecting path π. Since in G there is no transition node in M, π is active alternating with respect to M and $C \cup \mathrm{an}(C)$. If π has no collider node in $\mathrm{an}(C) \setminus C$ then by the nature of the construction of G we know that it has two edges (if both endpoints are children or parents) or three (if j is a parent and i a child or if the ij-edge is on a direction-preserving cycle) and that it has been generated by an edge (arrow, arc, or line) in H. If, for contradiction, there is a collider node $i \in \mathrm{an}(C) \setminus C$ on π then $i \in N_1$, and since the node in C has been generated by a line or an arrow on a direction-preserving cycle in H, $i \in \mathrm{an}(k)$ for a node k that is an endpoint of a line or is on a direction-preserving cycle. Hence H contains a ribbon, or the endpoints of the collider V-configuration with i as inner node are adjacent by an endpoint-identical edge. The former contradicts that H is ribbonless, and the latter contradicts that π is shortest.

b) We know that $\mathcal{SG} \subseteq \mathcal{RG}$. Notice that H is a summary graph. We know that, for the generated directed acyclic graph G, explained in (a), $\alpha_{RG}(G; M, C) = H$. Suppose, for contradiction, that step 2 of Algorithm 3.3 changes the graph. Thus a node with an arrowhead pointing to is in $\mathrm{an}(C)$, which implies that it is an ancestor of a node that is an endpoint of a line or on a direction-preserving cycle in H, a contradiction. Therefore, $\alpha_{SG}(G; M, C) = H$.

c) The result follows from part (b), the fact that $\mathcal{AG} \subseteq \mathcal{SG}$, and if $H \in \mathcal{AG}$ then $\alpha_{SG.AG}(H) = H$. □

We are now ready to depict the diagram in Figure 3.11 for the relationship between the three types of stable mixed graphs and DAGs.

3.5.3 Discussion on different types of stable mixed graphs

By what we discussed, if G is a DAG with latent variables M and selection variables C then stable mixed graphs are a class of graphs that represent the independence structure implied among the remaining variables, conditional on the selection variables. However, each of the three types has been used in different context and for different purposes.

Why MCGs or RGs? MCGs have been introduced in order to straightforwardly deal with the problem of finding a class of graphs that is closed under marginalisation and conditioning by a simple process of deriving them from DAGs. In fact, the class of MCGs is much larger than what one really needs for representing independence models after marginalisation and conditioning. We have noted that by using m-separation only MCGs that are ribbonless can be generated this way.

Why SGs? The main goal of defining SGs is to trace the effects after marginalisation and conditioning, as will be explained shortly in this section. By using binary matrix representations of graphs, called edge matrices, and corresponding matrix operators [Wermuth et al., 2006], the edge matrix of an SG is obtained. It contains three types of edge matrices: those for full lines, dashed lines (corresponding to arcs), and for arrows. In the family of joint Gaussian distributions, full lines in concentration graphs correspond to concentration matrices, dashed lines in covariance graphs to covariance matrices and arrows to equation parameters in structural equation models.

SGs are used when the generating DAG is known. Despite knowledge on the structure of the generating DAG, SGs are still of interest in two situations: (1) For models with large number of unobserved and selection variables; and (2) for the comparison of models when one of them has unobserved or selection variables

that are a subset of the unobserved or selection variables of the other.

Why AGs? Even though, we discussed the class of AGs in this thesis to sustain a parallel theory to RGs and SGs, the class of MAGs possess some desired properties that AGs do not. These include the fact that under the Gaussian path diagram parametrisation the MAG only implies independence constraints, while a general AG implies other types of constraint. MAGs are the simplest structures that capture the marginal and conditional independence models, and are also of interest when the generating DAG is not known, but a set of conditional independencies is known. In the Gaussian case MAGs are identified. In contrast to DAG models with hidden variables, the models are curved exponential families [Richardson and Spirtes, 2002], and conditional fitting algorithm for maximum likelihood estimation exists [Drton and Richardson, 2004].

As discussed, maximal AGs possess many desired properties that AGs do not. For SGs, it seems that MSGs possess the same statistical properties that both MAGs and SGs do possess. To show this, further work is needed.

The structure of different types of stable mixed graphs. If we suppose that stable mixed graphs are only used to represent the independence structure after marginalisation and conditioning, we can consider all types as equally appropriate. The question then will be reduced to how simple or fast generating a type of graph is. We have seen that AGs have the simplest structure among the three types of stable mixed graphs, and RGs are the most complex. However, we have also seen that it is more complex to generate an AG than to generate an SG, and to generate an SG than to generate an RG. On the other hand, the simpler structure allows a faster way of checking independence statements, i.e. applying the separation criterion. Hence it is a tradeoff that depends on the relative size of the marginalisation and conditioning sets in graphs.

When generating stable mixed graphs from DAGs, one always loses some information in order to obtain a simpler structure in stable mixed graphs. RGs have lost the least information among the three types of stable mixed graphs, while AGs the most. Here we discuss the lost information in the context of regression analysis.

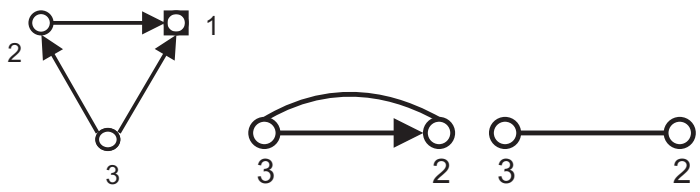
Multivariate regression and stable mixed graphs. The problem of constructing stable mixed graphs was originally posed by Wermuth et al. [1994] in the context of multivariate statistics based on regression analysis. In such literature the DAG model is defined by sequences of univariate recursive regressions, called a *linear triangular system* by Wermuth and Cox [2004], i.e. for $i = 1, \ldots, d_N - 1$, each single response variable Y_i is regressed on $Y_{\mathrm{pa}(i)}$, where the parents of i are a subset of $\{i+1, \ldots, d_N\}$. Linear triangular systems can be written as $AY = \epsilon$, where A is an upper-triangular matrix with unit diagonal elements, and ϵ is a vector of zero mean and uncorrelated random variables, called *residuals*. Here the non-zero regression coefficient of Y_i on Y_j can be attached the arrow from j to i in the DAG and is called the *direct effect* of Y_j on Y_i; see Cox and Wermuth [1993].

In particular, for linear triangular systems, RGs alert to distortions due to so-called over-conditioning via multiple edges consisting of a line and an arrow. Over-conditioning arises by conditioning on a variable that is a response of two variables, one of which itself is a response to the other one..

For example, in Fig. 3.12, the generating process is given by three linear equations,

$$Y_1 = \beta Y_2 + \delta Y_3 + \epsilon_1, \quad Y_2 = \gamma Y_3 + \epsilon_2, \quad Y_3 = \epsilon_3,$$

where each residual ϵ_i has mean zero and is uncorrelated with the explanatory variables on the right-hand side of the equation.



(a) (b) (c)

Figure 3.12: (a) A directed acyclic graph G with node 1 to be conditioned on. (b) RG generated from G. (c) SG or MAG generated from G.

By conditioning on Y_1, the conditional dependence of Y_2 on only Y_3 is obtained, which consists of the direct effect γ and an indirect effect of Y_2 on Y_3 via Y_1. This may be seen by direct calculation, assuming that the residuals ϵ_i have a Gaussian distribution, which leads to

$$E(Y_2 \,|\, Y_3) = (\gamma - \{(1-\gamma^2)/(1-\rho_{13}^2)\}\beta\rho_{13})Y_2, \text{ where } \rho_{13} = \delta + \beta\gamma.$$

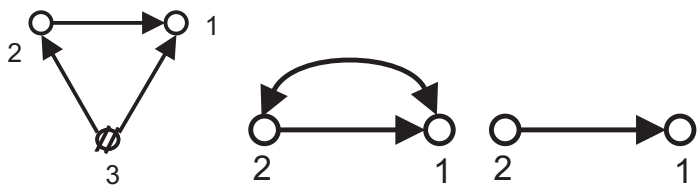
Thus, the direct effect γ is distorted by $-\{(1-\gamma^2)/(1-\rho_{13}^2)\}\beta\rho_{13}$. The potential presence of this distortion is represented in (b) by the presence of an arrow.

In addition, despite AGs, the existence of multiple edges with an arrow and an arc, and arcs with one endpoint ancestor of the other, respectively alerts distortions due to so-called direct and indirect *confounding*.

Indirect confounding was studied in Wermuth and Cox [2008] for marginalising only over a full set of background variables and also in Wermuth [2011] more generally relating SGs to corresponding maximal AGs. With the same generating process, as explained for Fig. 3.12, integrating out Y_3, the conditional dependence of Y_1 on only Y_2 is obtained, which consists of the direct effect β and an indirect effect of Y_1 on Y_2 via Y_3. This leads to

$$E(Y_1 \,|\, Y_2) = (\beta + \delta\gamma)Y_2.$$

Thus, the direct effect β is distorted by $\delta\gamma$. The potential presence of this distortion is represented in (b) by the presence of an arc. This example indicates a



(a) (b) (c)

Figure 3.13: (a) A directed acyclic graph G with node 3 to be marginalised over. (b) SG generated from G. (c) MAG generated from G.

distortion due to direct confounding; see Wermuth and Cox [2008] for examples of distortions due to *indirect confounding*.

3.6 Summary and future work

Summary. In this chapter by the extension of RG-generating algorithms we presented local algorithms to generate SGs and AGs and we developed a parallel theory for these three types of graphs to show that they are stable classes. We demonstrated the relationships between these graphs and discussed the pros and cons of the use of each type.

Future work. It is possible to demonstrate that the class of UCGs is not stable with respect to $\mathcal{J}_c(\mathcal{UCG})$, i.e. the class of chain graphs with the LWF Markov property is not stable under marginalisation. One natural extension of the results presented in this chapter is to close the c-connecting walks or, as named in Studeny [1998], superactive routes instead of m-connecting paths to generate a class of graphs that is closed under marginalisation over and conditioning on the node sets of LWF chain graphs. A parallel theory can then be developed by finding local algorithms to generate such an object, showing that the algorithm is well-defined, and proving that the generated class of graphs by the algorithm is stable.

Chapter 4

Loopless mixed graphs in R

4.1 Introduction

Goal and structure of the chapter. In this chapter we provide a manual that explains useful methods and functions for implementing the ideas described in previous chapters into the statistical computing programming language R.

First we explain different ways of defining DAGs (in Section 2) and mixed graphs (in Section 3) in R. Then, in Section 4, we introduce functions to generate different types of stable mixed graphs from DAGs or from graphs of the same class. These functions are basically the implemented generating functions as described in Chapter 3. In Section 5, we first provide a function to generate a maximal graph that induces the same independence model from a non-maximal graph. We also show how to verify the validity of given independence statements over the defined or generated graphs by using the m-separation criterion.

Preliminaries. The functions and codes outlined in this paper have been merged as a new set of functions into the R package `ggm`. For some previous contributions of `ggm`, see Marchetti [2006]. The package `ggm` has been improved and it is now more integrated with other contributed packages related to graphs, such as `graph`, `igraph` [Csardi and Nepusz, 2006], and `gRbase` [Dethlefsen and Højsgaard, 2005], which are now required for representing and plotting graphs.

In particular, in addition to adjacency matrices, all the functions in the package now accept `graphNEL` and `igraph` objects as input, as well as a new character string representation, which will be discussed later in this paper. Notice however that all the algorithms internally use the adjacency matrix representation.

A list of available packages in graphical models with some descriptions can be found at CRAN Task View gRaphical Models in R. This can be found at http://cran.r-project.org/web/views/gR.html.

First of all, we load the `ggm`-package.

```
> library(ggm)
```

4.2 Defining directed acyclic graphs

By "defining graphs" we mean defining the graph of interest as an input for the computer. There are several ways of defining DAGs. These are outlined as follows:

Defining DAGs as graphNEL objects. The `graph` package provides graphs in the name of `graphNEL` objects. These graphs can be either directed or undirected; there is thus the possibility of generating DAGs as `graphNEL` objects. The `gRbase` package provides the function `dag` for easily defining DAGs as `graphNEL` objects. On way in which the graph can be specified is by a list of formulas, so the DAG in Figure 1.2(a) can be defined by the following formula. For example, `~h*p*r` implies that h has parents p and r.

```
> exdag<-dag(~r,~q*r,~p,~h*p*r,~l,~k*l,~j*k,~i*h*l*j)
> exdag

A graphNEL graph with directed edges
Number of Nodes = 8
Number of Edges = 8
```

Notice that one cannot use integers as node labels in `graphNEL` objects. In addition, "*"s can be used instead of ":"s in the specification. In order to retrieve the nodes and edges of the DAG the functions `nodes` and `edges` are used:

```
> nodes(exdag)

[1] "r" "q" "p" "h" "l" "k" "j" "i"

> edges(exdag)

$r
[1] "q" "h"

$q
character(0)

$p
[1] "h"

$h
[1] "i"

$l
[1] "k" "i"

$k
[1] "j"

$j
[1] "i"
```

```
$i
character(0)
```

Plotting DAGs. In order to plot a DAG as a `graphNEL` object one can use the function `plot` in the `Rgraphviz` package. One can also use the various plotting options in the `igraph` package (such as the function `tkplot`) by changing the class of the DAG from `graphNEL` to `igraph` using the `igraph.from.graphNEL` function, but this function changes the node labels of the original DAG. The easiest way is, however, to use the `ggm` function `plotGraph`.

Defining DAGs as igraph objects. There are also many ways of defining DAGs as `igraph` objects. To see the full list of possible functions see Csardi and Nepusz [2006]. Probably the most handy way of generating small graphs is to use the function `graph.formula`. In this function we use "-" as edges and "+" as arrowheads. For example, for the same graph in Figure 1.2(a), we use the following formula:

```
> exdag<-graph.formula(v7+-v8-+v5+-v6,v5-+v1+-v4-+v3,v1+-v2+-v3)
> exdag

Vertices: 8
Edges: 8
Directed: TRUE
Edges:

[0] 'v8' -> 'v7'
[1] 'v8' -> 'v5'
[2] 'v6' -> 'v5'
[3] 'v5' -> 'v1'
[4] 'v4' -> 'v1'
```

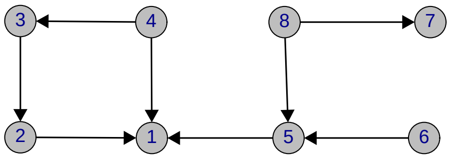
```
[5] 'v4' -> 'v3'
[6] 'v2' -> 'v1'
[7] 'v3' -> 'v2'
```

Defining DAGs by using adjacency matrices. The smartest way of defining DAGs is possibly to utilise their *adjacency matrix*. The adjacency matrix of a loopless directed graph $G = (V, E)$ is the matrix $A = (a_{ij})$ of size $|V| \times |V|$ with zeros on diagonal and $a_{ij} = \begin{cases} 1, & \text{if } i \dashrightarrow j; \\ 0, & \text{Otherwise.} \end{cases}$

One can use the function `graph.adjacency` to create a graph from an adjacency matrix in `igraph`. When applying the functions we are dealing with here, there is, however, no need to do so since these accept the adjacency matrices as a valid argument. For example, the same graph in Figure 1.2(a) can be created via:

```
> exvec<-rep(0,64)
> exmat<-matrix(exvec,8,8)
> exmat[8,7]<-1;exmat[8,5]<-1;exmat[6,5]<-1;exmat[5,1]<-1;exmat[4,1]<-
> + 1;exmat[4,3]<-1;exmat[2,1]<-1;exmat[3,2]<-1
> exmat
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,]    0    0    0    0    0    0    0    0
[2,]    1    0    0    0    0    0    0    0
[3,]    0    1    0    0    0    0    0    0
[4,]    1    0    1    0    0    0    0    0
[5,]    1    0    0    0    0    0    0    0
[6,]    0    0    0    0    1    0    0    0
[7,]    0    0    0    0    0    0    0    0
[8,]    0    0    0    0    1    0    1    0
```

```
> exdag<-graph.adjacency(exmat)
> plotGraph(exdag)
```



4.3 Defining mixed graphs

Goal of the section. None of the available packages in R has provided a tool for defining mixed graphs (and stable mixed graphs as subclasses). Such a tool must be able to provide graphs with multiple edges and edges of three different types. Here we introduce two methods to define mixed graphs to use as arguments of functions that generate stable mixed graphs from stable mixed graphs of the same type. These two methods also allow us to plot the graphs that have been defined using the generating functions. The first method uses a generalisation of the adjacency matrix described in the last section. The other one just uses a descriptive vector and is easy to use for small graphs.

Defining mixed graphs by using the adjacency matrices. We define the adjacency matrix related to a mixed graph as follows:

The matrix $A = (a_{ij})$ is called the *adjacency matrix* of the mixed graph $H = (N, F)$ if $|A| = |N| \times |N|$ and $A = B + S + W$, where for all $1 \leq i, j \leq |N|$

$$b_{ij} = \begin{cases} 1, & \text{if and only if } i \succ\!\!\!\longrightarrow j \text{ in } H; \\ 0, & \text{Otherwise.} \end{cases}$$

$$s_{ij} = s_{ji} == \begin{cases} 10, & \text{if and only if } i \text{ ——} j \text{ in } H; \\ 0, & \text{Otherwise.} \end{cases}$$

$$w_{ij} = w_{ji} = \begin{cases} 100, & \text{if and only if } i \prec\!\!\succ j \text{ in } H; \\ 0, & \text{Otherwise.} \end{cases}$$

Notice that because of the symmetric nature of lines and arcs S and W are symmetric, whereas B is not necessarily symmetric. Since there is a one-to-one correspondence between the set of mixed graphs and the set of adjacency matrices, one can easily generate the desired mixed graph by defining the correspondent adjacency matrix. All functions provided here can accept the adjacency matrices as valid arguments.

Defining mixed graphs by using vectors. There is another method for defining mixed graphs by using vectors. To define a mixed graph $H = (N, F)$ we use a vector of length $3|F|$ that is a sequence of triples ("type",node1 label,node2 label). The type of edge can be "a" (arrows from node1 to node2), "b" (arcs), and "l" (lines). This method does not create any isolated nodes. The two methods have been described in the following formulas:

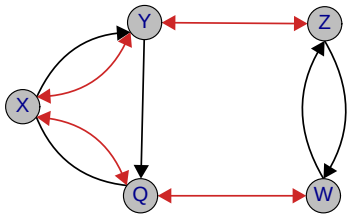
```
> mg <- matrix(c( 0, 101,   0,   0, 110,
+                100,   0, 100,   0,   1,
+                  0, 110,   0,   1,   0,
+                  0,   0,   1,   0, 100,
+                110,   0,   0, 100,   0),
+       5,5, byrow = TRUE)
> N <- c("X","Y","Z","W","Q")
> dimnames(mg) <- list(N, N)
> mg
    X   Y   Z   W   Q
X   0 101   0   0 110
Y 100   0 100   0   1
```

```
Z   0 110   0   1   0
W   0   0   1   0 100
Q 110   0   0 100   0
```

```
> mgv <- c("b","X","Y","a","X","Y","1","X","Q",
+          "b","Q","X","a","Y","Q","b","Y","Z",
+          "a","Z","W","a","W","Z","b","W","Q")
```

Plotting mixed graphs. Once again as in the DAG case we can use either `plotGraph(exmcmat)` or `plotGraph(exmcvec)` to plot the defined graph.

```
> plotGraph(mgv)
```



4.4 Generating stable mixed graphs

By "generating graphs" we mean applying the defined algorithms, e.g. those for generating stable mixed graphs, in order to generate new graphs.

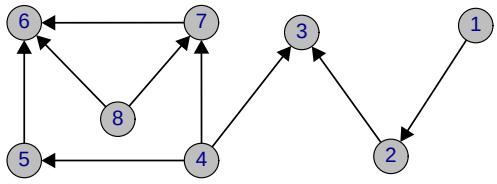
Three main functions to generate stable mixed graphs. Three main functions `RG`, `SG`, and `AG` are available to generate and plot ribbonless, summary, and ancestral graphs from DAGs, using the algorithms in Chapter 3.

The functions `RG`, `SG`, and `AG` all have three arguments: `a` the given input graph, `M`, the marginalization set and `C`, the conditioning set. The graph may be of class `graphNEL` or of class `igraph` or may be represented by a character vector, or by an adjacency matrix, as explained in the previous sections. The sets `M` and `C` must be disjoint vectors of node labels (default `c()`), and they may possibly be empty

sets. The output is always the adjacency matrix of the generated graph. There are two additional logical arguments `showmat` and `plot` to specify whether the adjacency matrix must be explicitly printed (default `TRUE`) and the graph must be plotted (default `FALSE`).

Some examples. We start from a DAG defined in two ways, as an adjacency matrix and as a character vector:

```
> ex <- matrix(c(0,1,0,0,0,0,0,0,
+                0,0,1,0,0,0,0,0,
+                0,0,0,0,0,0,0,0,
+                0,0,1,0,1,0,1,0,
+                0,0,0,0,0,1,0,0,
+                0,0,0,0,0,0,0,0,
+                0,0,0,0,0,1,0,0,
+                0,0,0,0,0,1,1,0),8,8,byrow=TRUE)
> exvec <- c("a",1,2,"a",2,3,"a",4,3,
+            "a",4,5,"a",4,7,"a",5,6,
+            "a",7,6,"a",8,6,"a",8,7)

> plotGraph(ex)
```



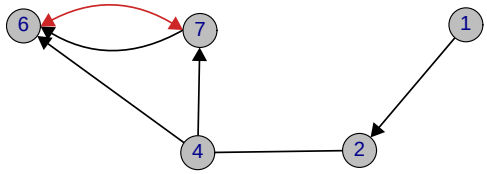
Then we define two disjoint sets M and C to marginalize over and condition on

```
> M <- c(5,8)
> C <- 3
```

and we generate the ribbonless, summary and ancestral graphs from the DAG with the associated plot.

```
> RG(ex, M, C, plot = TRUE)
```

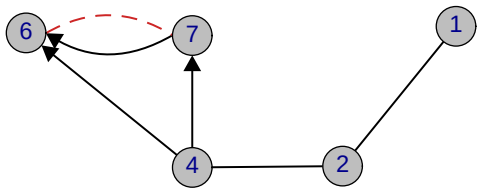
```
  1  2  4   6   7
1 0  1  0   0   0
2 0  0 10   0   0
4 0 10  0   1   1
6 0  0  0   0 100
7 0  0  0 101   0
```



The summary graph is plotted with dashed undirected edges instead of bi-directed edges:

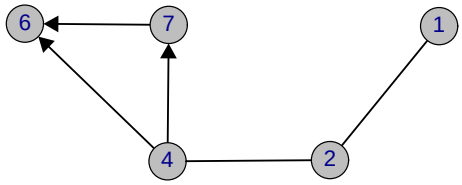
```
> plotGraph(SG(ex,M,C), dashed = TRUE)
```

```
   1  2  4   6   7
1  0 10  0   0   0
2 10  0 10   0   0
4  0 10  0   1   1
6  0  0  0   0 100
7  0  0  0 101   0
```



We illustrate hoe the induced ancestral graph may be obtained from the DAG defined as a vector.

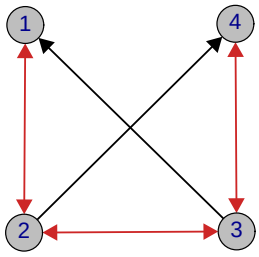
```
> AG(exvec,M,C,showmat=FALSE, plot=TRUE)
```



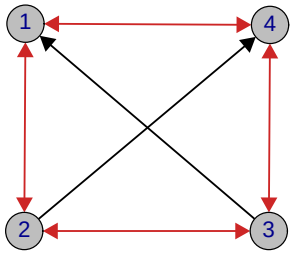
4.5 Maximality and m-separation

Function for generating maximal LMGs. Function `Max` gives the adjacency matrix of the generated maximal LMGs from non-maximal LMGs that induce the same independence model. This is equivalent to Algorithm 2.2. The related functions `MAG`, `MSG`, and `MRG`, are just handy wrappers to obtain maximal AGs, SGs and AGs, respectively. For example,

```
> H<-matrix(c(0,100,1,0,100,0,100,0,0,100,0,100,0,1,100,0),4,4)

> plotGraph(H)
```




```
> Max(H)

     1   2   3   4
1    0 100   0 100
2  100   0 100   1
3    1 100   0 100
4  100   0 100   0

> plotGraph(Max(H))
```



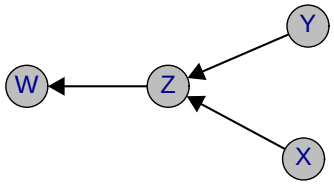
For example, this is the same as

```
> MAG(H,c(),c())
```

Function to verify m-separation. The verification of whether a graph implies an independence $A \perp\!\!\!\perp B|C$ may be done with the function `msep`. The function has 4 arguments, where the first is the graph `a`, in one of the forms discussed before, and the other three are the disjoint sets `A`, `B` and `C`. For example, for the following graph,

```
> a <- DAG(W ~ Z, Z ~ Y + X)
```

```
> plotGraph(a)
```



the two following statements verify whether `X` is m-separated from `Y` given `X`, and whether `X` is m-separated from `Y` (given the empty set):

```
> msep(a,"X","Y","Z")
```

```
[1] "NOT separated"
```

```
> msep(a,"X","Y")
```

```
[1] "Separated"
```

In this example m-separation is equivalent to its special case, the d-separation criterion for DAGs. (Note that there is another function `dSep` in `ggm` for d-separation.)

Glossary

armor the subgraph $\circ \!\!-\!\!\!- \circ \prec\!\!-\!\!\circ$ or $\circ \!\!-\!\!\!- \circ \prec\!\!\succ \circ$. 38

bow an arc with one endpoint that is the ancestor of the other endpoint. 39

chain a connected subgraph of a chain graph results by removing all arrows. 40

cycle a simple subgraph whose nodes can be placed around a circle so that two nodes are adjacent if these appear consecutively along the circle. 10

edge see graph. 8

- **multiple** two or more edges with the same pair of endpoints. 8

faithful probability distribution w.r.t. an independence model a joint probability distribution such that for every random vectors X_A, X_B, and X_C with this probability distribution, $A \perp\!\!\!\perp B \mid C$ if and only if $\langle A, B \mid C \rangle$ is in the independence model. 16

generating function a function $\alpha_{\mathcal{T}}$ from a class of graphs $\mathcal{T}$ to a stable class of graphs such that for a graph $H_2 \in \mathcal{T}$ for which $\mathcal{J}^{H_2} = \alpha(\mathcal{J}_m(H_1); M, C)$, $H_2 = \alpha_{\mathcal{T}}(H_1; M, C)$. 73

graph a triple that consists of a node set, an edge set, and a relation that with each edge associates two nodes. 8

- **acyclic** a graph with no direction-preserving cycle. 11
- **acyclic armorless** a mixed graph that contains no armor and no direction-preserving cycle. 38
- **acyclic directed mixed** a loopless mixed graph that consists only of arcs and arrows and contains no direction-preserving cycles. 39

graph (continued)

ancestral a simple mixed graph that has the following properties for all nodes i: (1) $i \notin \mathrm{an}(\mathrm{pa}(i) \cup \mathrm{sp}(i))$; (2) if $\mathrm{ne}(i) \neq \varnothing$ then $\mathrm{pa}(i) \cup \mathrm{sp}(i) = \varnothing$. 39

anterior graph of the graph obtained from a graph by repeatedly removing arrowheads pointing to nodes that are the endpoints of a line and considering only one edge of the same type between each pair of nodes. 23

bidirected a graph that only consists of arcs. 42

chain a graph with two types of edges (arrows and one symmetric type of edge, which can be lines or arcs) such that it does not contain any cycle that contains at least one arrow and may contain symmetric edges and whose arrows are all pointing to one direction. 40

bidirected a chain graph with chain components connected by arcs. 41

undirected a chain graph with chain components connected by lines. 40

concentration an undirected graph. 42

connected a graph such that there exists a path between any pairs of nodes. 10

corresponding ribbonless, summary, and ancestral graphs $\alpha_{RG}(G; M, C)$, $\alpha_{SG}(G; M, C)$, and $\alpha_{AG}(G; M, C)$ for a directed acyclic graph G and two disjoint subsets of its node set M and C. 115

covariance a bidirected graph. 42

directed a graph that only consists of arrows. 9

independence a maximal graph. 45

maximal w.r.t. a separation criterion a graph such that by adding any edge to it the independence model induced by the separation criterion changes. 45

MC a mixed graph. 31

mixed a graph that contains three types of edges denoted by arrows, arcs,

and lines. 19

loopless a mixed graph that contains no loops. 19

graph (continued)

multivariate regression chain a bidirected chain graph. 41

partial-ancestor w.r.t. a set of nodes the graph generated by turning every ancestor whose inner nodes are in the set into a parent. 11

ribbonless a loopless mixed graph that does not contain ribbons. 28

simple a graph with no loops or multiple edges. 8

stable class of w.r.t. a class of independence models corresponding to graphs in the class a class of graphs such that for every graph G in the class and every disjoint subsets M and C of the node set of G, there is a graph H in the class such that $\mathcal{J}^H = \alpha(\mathcal{J}^G; M, C)$. 69

subgraph of a graph whose node set and edge set are subsets of the node set and the edge set of the graph. 9

induced by a set of nodes or edges a subgraph that contains all and only nodes or edges in A and all edges between two nodes in A. 9

summary an acyclic armorless graph. 38

undirected a graph that consists only of lines. 42

graphoid an independence model that satisfies semigraphoid axioms plus the intersection property. 15

compositional an independence model that satisfies graphoid axioms plus the composition property. 15

semi an independence model that satisfies symmetry, decomposition, weak union, and contraction properties. 14

independence model over a set a set of independence statements from the set. 7

after marginalisation over set of nodes M and conditioning on set of nodes C independence model $\{\langle A, B \mid D\rangle : \langle A, B \mid D \cup C\rangle \in \mathcal{J}$ and $(A \cup$

$B \cup D) \cap (M \cup C) = \varnothing\}$. 68

independence model over a set (continued)

probabilistic an independence model w.r.t. which there is a distribution that is faithful. 16

satisfying a pairwise Markov property w.r.t. function h an independence model $\mathcal{J}$ such that $i \not\sim j$ implies $\langle i, j \,|\, C\rangle \in \mathcal{J}$, for $C = h(i, j)$. 13

satisfying the global Markov property w.r.t. separation criterion $\mathcal{S}$ an independence model $\mathcal{J}$ such that $A \perp_{\mathcal{S}} B \,|\, C$ implies $\langle A, B \,|\, C\rangle \in \mathcal{J}$. 13

independence statement a triple $\langle X, Y \,|\, Z\rangle$, interpreted as "$X$ is independent of Y given Z". 7

loop an edge with the same endpoints. 8

node see graph. 8

adjacent two nodes that are the endpoints of an edge. 8

ancestor of an endpoint of a direction-preserving path pointing to the node. 11

A**-line** ancestor of a node such that all inner nodes in the direction-preserving path are in set A. 11

anterior of a node from which there is an anterior path to the node in the anterior graph. 23

child of a node to which there is an arrow pointing from the node. 9

collider the inner node of three V-configurations $i \rightarrow t \leftarrow j$, $i \leftrightarrow t \leftarrow j$, and $i \leftrightarrow t \leftrightarrow j$. 22

collision node c on $i \rightarrow c \leftarrow j$. 10

degree of the number of edges adjacent to the node. 8

descendant of nodes along the direction-preserving path from the node. 11

neighbour of a node adjacent to the node by a line. 22

node (continued)

non-collider the inner node of all V-configurations but the ones for collider V-configurations. 22

parent of a node that has an arrow pointing to the node. 9

source node s on $i \leftarrow s \rightarrow j$. 10

spouse of a node adjacent to the node by an arc. 22

transition node t on $i \leftarrow t \leftarrow j$. 10

path a walk with no repeated node or edge. 9

active w.r.t. two sets of nodes A and B a path whose source nodes are in A and collision nodes are in B. 11

alternating a path whose direction of arrows changes at each inner node. 11

active w.r.t. a set of nodes an edge or an alternating path with some inner nodes such that all its source nodes are in the set and all its collision nodes are not in the set. 11

anterior path a path from i to j in the anterior graph that has the form $i - i_1 - \ldots - i_m \rightarrow i_{m+1} \rightarrow \ldots \rightarrow j$. 23

combination of The path $\pi_{12} = \pi_1 \circ \pi_2 = \langle i, \ldots, k, j_{q-1}, \ldots, j \rangle$ for two paths $\pi_1 = \langle i = i_0, i_1, \ldots, i_n, h \rangle$ and $\pi_2 = \langle h, j_m, j_{m-1}, \ldots, j_0 = j \rangle$, where $k = i_p = j_q$ is the first node of π_1 on both paths. 9

d-connecting given a set of nodes a path along which every collision node is in the set or has a descendant in the set and every other node is outside the set. 43

direction-preserving a path whose inner nodes are transition nodes. 11

endpoint-identical Two paths π_1 and π_2 between i and j such that there is an arrowhead pointing to i in π_1 if and only if there is an arrowhead pointing to i in π_2 and similarly for j. 22

endpoints of the first and the last nodes of the path. 10

inducing w.r.t. sets of nodes C and M a path whose collider nodes are

in C or in $\mathrm{an}(\{i,j\} \cup C)$ and non-collider nodes are in M. 100

non-ancestral an inducing path $\langle i = i_0, i_1, \ldots, i_m = j \rangle$ that can be partitioned into two subpaths $\langle i, i_1, \ldots, i_l \rangle$ and $\langle i_{l+1}, \ldots, i_m \rangle$ (where one path can be empty) such that i_n, $1 \leq n \leq l$, that are collider are in $C \cup \mathrm{an}(C \cup \{i\})$ and i_p, $l+1 \leq p \leq m-1$, that are collider are in $C \cup \mathrm{an}(C \cup \{j\})$. 102

path (continued)

inner node of a node that is on the path and not an endpoint of that path. 10

m-connecting given a set of nodes a path whose collider nodes are in the set of nodes or its ancestors and all its non-collider nodes are outside the set of nodes. 25

m-connecting given sets of nodes M and C a path along which every collider node is in $C \cup \mathrm{ant}(C)$ and every non-collider node is in M. 74

primitive inducing a path between i and j such that, for every n, $1 \leq n \leq p$, (i) q_n is a collider on the path; and (ii) $q_n \in \mathrm{an}(\{i\} \cup \{j\})$. 46

ancestral primitive inducing a path $\pi = \langle i_0, i_1, \ldots, i_{n-1}, i_n \rangle$ such that it consists only of arcs, $i_1 \in \mathrm{an}(i_n)$, $i_{n-1} \in \mathrm{an}(i_0)$, and for $k \in \{2, \ldots, n-2\}$, either $i_k \in \mathrm{an}(i_0)$ or $i_k \in \mathrm{an}(i_n)$. 54

subpath a path that can be considered a subgraph of the path with the ordering associated with that path. 10

property

composition if $\langle A, B \,|\, C \rangle \in \mathcal{J}$ and $\langle A, D \,|\, C \rangle \in \mathcal{J}$ then $\langle A, B \cup D \,|\, C \rangle \in \mathcal{J}$. 15

contraction $\langle A, B \,|\, C \cup D \rangle \in \mathcal{J}$ and $\langle A, D \,|\, C \rangle \in \mathcal{J}$ if and only if $\langle A, B \cup D \,|\, C \rangle \in \mathcal{J}$. 14

decomposition if $\langle A, B \cup D \,|\, C \rangle \in \mathcal{J}$ then $\langle A, B \,|\, C \rangle \in \mathcal{J}$ and $\langle A, D \,|\, C \rangle \in \mathcal{J}$. 14

property (continued)

intersection if $\langle A, B \,|\, C \cup D\rangle \in \mathcal{J}$ and $\langle A, D \,|\, C \cup B\rangle \in \mathcal{J}$ then $\langle A, B \cup D \,|\, C\rangle \in \mathcal{J}$. 15

symmetry $\langle A, B \,|\, C\rangle \in \mathcal{J}$ if and only if $\langle B, A \,|\, C\rangle \in \mathcal{J}$. 14

weak union if $\langle A, B \cup D \,|\, C\rangle \in \mathcal{J}$ then $\langle A, B \,|\, C \cup D\rangle \in \mathcal{J}$ and $\langle A, D \,|\, C \cup B\rangle \in \mathcal{J}$. 14

ribbon a collider V-configuration $\langle h, i, j\rangle$ such that (1) there is no endpoint-identical edge between h and j; (2) i or a descendant of i is the endpoint of a line or a direction-preserving cycle.. 28

loop-ribbon a ribbon $\langle i, k, j\rangle$ with k being the endpoint of or being an ancestor of a node that is the endpoint of a line loop. 32

non-loop-ribbon a ribbon that is not a loop-ribbon. 32

separation criterion a function $\mathcal{S}$ that maps $G = (V, E)$ to an independence model $\mathcal{J}_{\mathcal{S}}(G)$ and has the two following properties: (1) it is path-based; (2) if there are nodes $i \in A$ and $j \in B$ such that $i \sim j$ then $\langle A, B \,|\, C\rangle \notin \mathcal{J}_{\mathcal{S}}(G)$. 11

V-configuration a path with three nodes and two edges. 10

collider a V-configuration with inner collider node. 22

collision a V-configuration with inner collision node. 10

non-collider a V-configuration with inner non-collider node. 22

non-collision a transitions or source V-configuration. 10

source a V-configuration with inner source node. 10

transition a V-configuration with inner transition node. 10

unshielded a V-configuration whose endpoints are not adjacent. 10

walk a list $\langle v_0, e_1, v_1, \ldots, e_k, v_k\rangle$ of nodes and edges such that for $1 \leq i \leq k$, the edge e_i has endpoints v_{i-1} and v_i. 9

blocked by a set of nodes C a walk that is not open given C. 32

walk (continued)

open given a set of nodes C a walk for which $C_\pi \subseteq C$ and $N_\pi \subseteq N \setminus C$, where $C_\pi := \{j \in N : j \text{ occurs as a collider on } \pi\}$ and $N_\pi := \{j \in N : j \text{ occurs as a non-collider on } \pi\}$. 32

Bibliography

S. A. Andersson and M. D. Perlman. Characterizing Markov equivalence classes for AMP chain graph models. *The Annals of Statistics*, 34(2):939–972, 2006.

J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society, Series B*, 36(2):192–236, 1974.

H. M. Blalock, editor. *Causal Models in the Social Sciences*. Macmillan, London, United Kingdom, 1971.

P. Clifford. Markov random fields in statistics. In G. R. Grimmett and D. J. A. Welsh, editors, *Disorder in Physical Systems: A Volume in Honour of John M. Hammersley*, pages 19–32. Oxford University Press, 1990.

D. R. Cox and N. Wermuth. Linear dependencies represented by chain graphs (with discussion). *Statistical Science*, 8(3):204–218; 247–277, 1993.

D. R. Cox and N. Wermuth. *Multivariate Dependencies : models, analysis and interpretation*. Chapman & Hall, London, United Kingdom, 1996.

G. Csardi and T. Nepusz. The `igraph` software package for complex network research. *InterJournal*, Complex Systems:1695, 2006. URL `http://igraph.sf.net`.

J. N. Darroch, S. L. Lauritzen, and T. P. Speed. Markov fields and log-linear interaction models for contingency tables. *Annals of Statistics*, 8:522–539, 1980.

A. P. Dawid. Conditional independence in statistical theory (with discussion). *Journal of the Royal Statistical Society Series B*, 41(1):1–31, 1979.

A. P. Dempster. Covariance selection. *Biometrics*, 28(1):157–175, 1972.

C. Dethlefsen and S. Højsgaard. A common platform for graphical models in R: The gRbase package. *Journal of Statistical Software*, 14(17):1–12, 12 2005. ISSN 1548-7660. URL `http://www.jstatsoft.org/v14/i17`.

M. Drton. Discrete chain graph models. *Bernoulli*, 15(3), 2009.

M. Drton and T. Richardson. Iterative conditional fitting for Gaussian ancestral graph models. In *Proceedings of the Proceedings of the Twentieth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-04)*, pages 130–137, Arlington, Virginia, 2004. AUAI Press.

M. Frydenberg. The chain graph Markov property. *Scandinavian Journal of Statistics*, 17:333–353, 1990.

D. Geiger, T. S. Verma, and J. Pearl. Identifying independence in Bayesian networks. *Networks*, 20:507–534, 1990.

J. W. Gibbs. *Elementary Principles of Statistical Mechanics*. Yale University Press, New Haven, Connecticut, USA, 1902.

L. A. Goodman. The multivariate analysis of qualitative data: interaction among multiple classifications. *Journal of the American Statistical Association*, 66: 339–344, 1970.

G. R. Grimmett. A theorem about random fields. *Bulletin of the London Mathematical Society*, 5:81–84, 1973.

J. M. Hammersley and P. Clifford. Markov fields on finite graphs and lattices. Unpublished, 1971.

V. Hodapp and N. Wermuth. Decomposable models: A new look at interdependence and dependence structures in psychological research. *Multivariate Behavioral Research*, 18:361–390, 1983.

C. Kang and J. Tian. Markov properties for linear causal models with correlated errors. *Journal of Machine Learning Research*, 10:41–70, 2009.

G. Kauermann. On a dualization of graphical Gaussian models. *Scandinavian Journal of Statistics*, 23(1):105–116, 1996.

H. Kiiveri, T. P. Speed, and J. B. Carlin. Recursive causal models. *Journal of the Australian Mathematical Society, Series A*, 36:30–52, 1984.

J. T. A. Koster. Marginalizing and conditioning in graphical models. *Bernoulli*, 8(6):817–840, 2002.

S. L. Lauritzen. *Graphical Models*. Clarendon Press, Oxford, United Kingdom, 1996.

S. L. Lauritzen and N. Wermuth. Graphical models for association between variables, some of which are qualitative and some quantitative. *Annals of Statistics*, 17:31–57, 1989.

S. L. Lauritzen, A. P. Dawid, B. N. Larsen, and H. G. Leimer. Independence properties of directed Markov fields. *Networks*, 20:491–505, 1990.

G. M. Marchetti. Independencies induced from a graphical Markov model after marginalization and conditioning: the R package ggm. *Journal of Statistical Software*, 15(6), 2006.

G. M. Marchetti and N. Wermuth. Matrix representation and independencies in directed acyclic graphs. *Annals of Statistics*, 37(2):961–978, 2009.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems : networks of plausible inference*. Morgan Kaufmann Publishers, San Mateo, CA, USA, 1988.

J. Pearl and A. Paz. Graphoids: a graph based logic for reasoning about relevancy relations. *Advances in Artificial Intelligence-II*, pages 357–363, 1987.

J. A. Rice. *Mathematical Statistics and Data Analysis*. Thomson Brooks/Cole, Belmont, CA, USA, 2007.

T. S. Richardson and P. Spirtes. Ancestral graph Markov models. *Annals of Statistics*, 30(4):962–1030, 2002.

K. Sadeghi. Stable mixed graphs. *Bernoulli*, To appear, 2012. URL `http://arxiv.org/abs/1110.4168`.

K. Sadeghi and S. L. Lauritzen. Markov properties for mixed graphs. *submitted*, 2011. URL `http://arxiv.org/abs/1109.5909`.

K. Sadeghi and G. Marchetti. Graphical Markov models with mixed graphs in R. *submitted*, 2011.

P. Spirtes, T. Richardson, and C. Meek. The dimensionality of mixed ancestral graphs. Technical Report CMU-PHIL-83, Philosophy Department, CMU, 1997.

F. Spitzer. *Principles of Random Walks (Graduate Texts in Mathematics)*. Springer-Verlag, New York-Heidelberg, 1971.

M. Studeny. Multiinformation and the problem of characterization of conditional independence relations. *Problems of Control and Information Theory*, 18:3–16, 1989.

M. Studeny. Bayesian networks from the point of view of chain graphs. In *UAI*, pages 496–503, San Francisco, CA, 1998. Morgan Kaufmann.

M. Studeny. *Probabilistic Conditional Independence Structures*. Springer-Verlag, London, United Kingdom, 2005.

M. Studeny and R. R. Bouckaert. On chain graph models for description of conditional independence structures. *The Annals of Statistics*, 26(4):1434–1495, 1998.

N. Wermuth. Probability distributions with summary graph structure. *Bernoulli*, 17(3):845–879, 2011.

N. Wermuth and D. R. Cox. On association models defined over independence graphs. *Bernoulli*, 4:477–495, 1998.

N. Wermuth and D. R. Cox. Joint response graphs and separation induced by triangular systems. *J. Roy. Statistical Society B*, 66:687–717, 2004.

N. Wermuth and D. R. Cox. Distortion of effects caused by indirect confounding. *Biometrika*, 95:17–33, 2008.

N. Wermuth and K. Sadeghi. Sequences of regressions and their independences. *TEST*, To appear as an invited discussion paper, 2011. URL `http://arxiv.org/abs/1103.2523`.

N. Wermuth, D. R. Cox, and J. Pearl. Explanation for multivariate structures derived from univariate recursive regressions. Technical Report 94(1), Univ. Mainz, Germany, 1994.

N. Wermuth, M. Wiedenbeck, and D. R. Cox. Partial inversion for linear systems and partial closure of independence graphs. *BIT Numerical Mathematics*, 46: 883–901, 2006.

N. Wermuth, G. M. Marchetti, and D. R. Cox. Triangular systems for symmetric binary variables. *Electronic Journal of Statistics*, 3:932–955, 2009.

D. B. West. *Introduction to Graph Theory*. Prentice Hall, Upper Saddle River, NJ, USA, 2001.

H. D. A. Wold. Causality and econometrics. *Econometrica*, 22(2):162–177, 1954.

S. Wright. The method of path coefficients. *Annals of Mathematics Statistics*, 5 (3):161–215, 1934.

H. Zhao, Z. Zheng, and B. Liu. On the Markov equivalence of maximal ancestral graphs. *Science in China Ser. A, Mathematics*, 48:548–562, 2004.