Understanding the Scalability of Bayesian Network Inference using Clique Tree Growth Curves*

Ole J. Mengshoel
Carnegie Mellon University
NASA Ames Research Center
Mail Stop 269-3
Moffett Field, CA 94035
Ole.Mengshoel@sv.cmu.edu

Abstract

One of the main approaches to performing computation in Bayesian networks (BNs) is clique tree clustering and propagation. The clique tree approach consists of propagation in a clique tree compiled from a Bayesian network, and while it was introduced in the 1980s, there is still a lack of understanding of how clique tree computation time depends on variations in BN size and structure. In this article, we improve this understanding by developing an approach to characterizing clique tree growth as a function of parameters that can be computed in polynomial time from BNs, specifically: (i) the ratio of the number of a BN's non-root nodes to the number of root nodes, and (ii) the expected number of moral edges in their moral graphs. Analytically, we partition the set of cliques in a clique tree into different sets, and introduce a growth curve for the total size of each set. For the special case of bipartite BNs, there are two sets and two growth curves, a mixed clique growth curve and a root clique growth curve. In experiments, where random bipartite BNs generated using the BPART algorithm are studied, we systematically increase the out-degree of the root nodes in bipartite Bayesian networks, by increasing the number of leaf nodes. Surprisingly, root clique growth is well-approximated by Gompertz growth curves, an S-shaped family of curves that has previously been used to describe growth processes in biology, medicine, and neuroscience. We believe that this research improves the understanding of the scaling behavior of clique tree clustering for a certain class of Bayesian networks; presents an aid for trade-off studies of clique tree clustering using growth curves; and ultimately provides a foundation for benchmarking and developing improved BN inference and machine learning algorithms.

1 Introduction

Bayesian networks (BNs) play a central role in a wide range of automated reasoning applications, including in diagnosis, sensor validation, probabilistic risk analysis, information fusion, and decoding of error-correcting codes [64, 6, 59, 38, 37, 60, 43, 58]. A crucial issue in reasoning using BNs, as well as in other forms of model-based reasoning, is that of computational scalability. Most BN inference problems are computationally hard in the general case [10, 63, 61, 1], thus there may be reason to be concerned about scalability. One can make progress on the scalability question by studying classes of problem instances analytically and experimentally. Such problem instances may come from applications or they may be randomly generated. In the area of application BNs, both encouraging and discouraging scalability results have been reported. For example, a prominent bipartite BN for medical diagnosis is known to be intractable using current technology [64]. Decoding of error-correcting codes, which can be understood as BN inference, is also not tractable but has empirically been found to be solvable with high reliability using inexact BN inference [20, 37]. On the other hand, it is well-known that BNs that are tree-structured, including the so-called naive Bayes model, are solvable in polynomial time using exact inference algorithms. There are also encouraging empirical results for application BNs that are "close" to being tree-structured or more generally for application BNs that are not highly connected [26, 43].

Clique tree clustering, where inference takes the form of propagation in a clique tree compiled from a BN, is currently among the most prominent BN inference algorithms [33, 2, 62]. The performance of tree clustering algorithms depends on a BN's treewidth or the optimal maximal clique size of a BN's induced clique tree [16, 11, 15]. The performance of other exact BN inference algorithms also depends on treewidth. A key research question is, then, how

*Accepted for publication in the Artificial Intelligence Journal.

the size of a clique tree generated from a BN (and consequently, inference time) depends on structural measures of BNs. One way to investigate this is through the use of random generation from distributions of problem instances [66, 5, 11, 52, 23]. Taking this approach, and increasing the ratio C/V between the number of leaf nodes C and the number of root nodes V in bipartite BNs, an easy-hard-harder pattern along with approximately exponential growth have previously been observed for clique tree clustering for a certain class of BNs, namely BPART BNs [45].

In this article, we develop a more precise understanding of this easy-hard-harder pattern. This is done by formulating macroscopic and approximate models of clique tree growth by means of restricted growth curves, which we illustrate by using bipartite BNs created by the BPART algorithm [45]. For the sake of this work, we assume that a clique tree propagation algorithm, operating on a clique tree compiled from a BN, is executed in order to answer probabilistic queries of interest. We introduce a random variable for total clique tree size. This random variable is, for the case of bipartite BNs, the sum of two random variables, one for the size of root cliques and one for the size of mixed cliques. Reflecting the random variable for total clique tree size, we introduce a continuous growth curve for total clique tree size which is the sum of growth curves for the size of root cliques and mixed cliques. Of particular interest is the growth curve for root clique size, where Gompertz curves of the form $g(\infty)e^{-\zeta e^{-\gamma x}}$, where $g(\infty)$, ζ, and γ are parameters, turn out to be useful. A key finding is that Gompertz growth curves are justified on theoretical grounds and also fit very well to experimental data generated using the BPART algorithm [45]. While we emphasize bipartite BNs in this article, we also discuss how to generalize to arbitrary BNs, by using multiple growth curves or translating arbitrary BNs to bipartite BNs via factor graphs [32, 70].

For experimentation, we sampled bipartite BNs using an implementation of the BPART algorithm. For the number of root nodes, V, we used $V = 20$ and $V = 30$. The number of leaf nodes was also varied, thereby creating BNs of varying hardness; 100 BNs per C/V-level were randomly generated. A clique tree inference system, employing the minimum fill-in weight heuristic, was used to generate clique trees for the sampled BNs. Let W be a random variable representing the number of moral edges in moral graphs induced by random BNs. In addition to $x = C/V$, we consider $x = E(W)$ as an independent variable. In experiments, we compared different growth curves and investigated $x = C/V$ versus $x = E(W)$ as independent variables for Gompertz growth curves. Linear regression was used to obtain values for the parameters ζ and γ based on a linear form of the Gompertz growth curve; values for $g(\infty)$ were obtained by analysis. Gompertz growth curves are common in biological, medical, and neuroscience research [4, 35, 17], but have not previously been used to characterize clique tree growth (except for in our earlier conference paper [41] which this article extends). We provide improved results compared to previous research, where an easy-hard-harder pattern and approximately exponential growth of upper bounds on optimal maximal clique size as a function of C/V-ratio were established [45].

We believe this research is significant for the following reasons. First, analytical growth curves improve the understanding of clique tree clustering's performance for a certain class of BNs, namely BPART BNs. Consider Kepler's three laws of planetary motion, developed using Brahe's observational data of planetary movement. There is a need to develop similar laws for clique tree clustering's performance, and in this article we obtain such laws in the form of Gompertz growth curves for BPART BNs [45]. While they admittedly have a strong empirical basis, these Gompertz growth curves give significantly better fit to the raw data than alternative curves. Consequently, they provide better insight into the underlying mechanisms of the clique tree clustering algorithm and may be used to approximately predict the performance of the algorithm. Since the performance of other exact BN inference algorithms – including conditioning [55, 11] and elimination algorithms [34, 71, 14] – also depends on optimal maximal clique size, our results may have significance for these algorithms as well. A second benefit of growth curves is that they can be used to summarize performance of different BN inference algorithms or different implementations of the same algorithm on benchmark sets of problem instances, and thereby aid in evaluations.[1] Suppose that the growth curves $g_1(x)$ and $g_2(x)$ were obtained by benchmarking slightly different clique tree algorithms. Compared to looking at and evaluating potentially large amounts of raw data, it may be easier to understand the performance difference between the two algorithms by studying their curves $g_1(x)$ and $g_2(x)$ or by comparing their respective Gompertz curve parameter values ζ_1 and γ_1 versus ζ_2 and γ_2. A third benefit is that growth curves provide estimates of resource consumption in terms of clique tree size, estimates that can easily be translated into requirements on memory size and inference time. Hence, this approach enables trade-off studies of resource consumption (or requirements) versus resource bounds, which is important in resource-bounded reasoners [48, 40], and may also be of use if one wants to take into account, during BN structure learning, the computational resources needed for reasoning.

The rest of this article is organized as follows. After introducing notation and background concepts related to Bayesian networks and clique tree clustering in Section 2, we study, in the context of the BPART algorithm, the development and growth of BNs, causing corresponding clique tree growth. Specifically, in Section 3 we study the

[1] Such evaluations have been performed, for example, at recent UAI conferences, see `http://ssli.ee.washington.edu/~bilmes/uai06InferenceEvaluation/` and `http://graphmod.ics.uci.edu/uai08/` for details. Application BNs for benchmarking can be found at `http://genie.sis.pitt.edu/networks.html` and `http://www.cs.huji.ac.il/labs/compbio/Repository/`.

issue of independent variables for growth curves, in particular the C/V-ratio and the expected number of moral edges $E(W)$, and discuss how growth curves can provide a macroscopic model of how clique trees grow. In Section 4, we discuss the connection between random graphs and our BPART model. In Section 5, we present experiments with BNs generated using the BPART algorithm, varying the number of root and leaf nodes. We compare different mathematical models of growth, and find that Gompertz growth curves give the best fit to sample data. We conclude and indicate future research directions in Section 6.

2 Background

Graphs, and in particular directed acyclic graphs as introduced in the following definition, play a key role in Bayesian networks.

Definition 1 (Directed acyclic graph (DAG)) *Let $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ be a non-empty directed acyclic graph (DAG) with nodes $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ for $n \geq 1$ and edges $\boldsymbol{E} = \{E_1, \ldots, E_m\}$ for $m \geq 0$. An ordered tuple $E_i = (Y, X)$, where $0 \leq i \leq m$ and $X, Y \in \boldsymbol{X}$, represents a directed edge from Y to X. Π_X denotes the parents of X: $\Pi_X = \{Y \mid (Y, X) \in \boldsymbol{E}\}$. Similarly, Ψ_X denotes the children of X: $\Psi_X = \{Z \mid (X, Z) \in \boldsymbol{E}\}$, where $Z \in \boldsymbol{X}$. The out-degree and in-degree of a node X are defined as $o(X) = |\Psi_X|$ and $i(X) = |\Pi_X|$ respectively.*

In the rest of this article we assume for simplicity that DAGs and BNs are non-empty, even when not explicitly stated as in Definition 1. The following classification of nodes in DAGs, including in BNs, turns out to be useful when we discuss the performance of BN inference algorithms.

Definition 2 *Let $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ be a non-empty DAG with $X \in \boldsymbol{X}$. If $i(X) = 0$ then X is a root node. If $i(X) > 0$ then X is a non-root node. If $i(X) > 0$ and $o(X) = 0$ then X is a leaf node. If $o(X) > 0$ then X is a non-leaf node. If $o(X) > 0$ and $i(X) > 0$ then X is a trunk (non-leaf and non-root) node .*

With the concepts from Definition 2 in hand, we classify the nodes in a DAG as follows.

Definition 3 *Let $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ be a DAG. We identify the following subsets of $\boldsymbol{X}$: $\boldsymbol{V} = \{X \in \boldsymbol{X} \mid i(X) = 0\}$ (the root nodes); $\boldsymbol{C} = \{X \in \boldsymbol{X} \mid i(X) > 0$ and $o(X) = 0\}$ (the leaf nodes); $\boldsymbol{T} = \{X \in \boldsymbol{X} \mid i(X) > 0$ and $o(X) > 0\}$ (the trunk nodes); $\bar{\boldsymbol{V}} = \{X \in \boldsymbol{X} \mid i(X) > 0\}$ (the non-root nodes); and $\bar{\boldsymbol{C}} = \{X \in \boldsymbol{X} \mid o(X) > 0\}$ (the non-leaf nodes).*

A Bayesian network (BN) is a DAG with an associated set of conditional probability distributions [56]. In the following definition, we let $n = |\boldsymbol{X}|$ and let π_{X_i} represent a complete instantiation of the parents Π_{X_i} of X_i, in other words $\pi_{X_i} \subseteq \boldsymbol{x}$ is a projection of a complete assignment $\boldsymbol{x} = \{X_1 = x_1, \ldots, X_n = x_n\}$.

Definition 4 (Bayesian network) *A Bayesian network is a tuple $\beta = (\boldsymbol{X}, \boldsymbol{E}, \boldsymbol{P})$, where $(\boldsymbol{X}, \boldsymbol{E})$ is a DAG augmented with conditional probability distributions $\boldsymbol{P} = \{\Pr(X_1 \mid \Pi_{X_1}), \ldots, \Pr(X_n \mid \Pi_{X_n})\}$. Here, $\Pr(X_i \mid \Pi_{X_i})$ is the conditional probability distribution for $X_i \in \boldsymbol{X}$. The independence assumptions encoded in $(\boldsymbol{X}, \boldsymbol{E})$ imply the joint probability distribution*

$$\Pr(\boldsymbol{x}) = \Pr(x_1, \ldots, x_n) = \Pr(X_1 = x_1, \ldots, X_n = x_n) = \prod_{i=1}^{n} \Pr(x_i \mid \pi_{X_i}). \tag{1}$$

In this article we will restrict ourselves to discrete random variables, and "BN node" will thus mean "discrete BN node". Let a BN node $X \in \boldsymbol{X}$ have states $\{x_1, \ldots, x_m\}$. We use the notation $\Omega_X = \Omega(X) = \{x_1, \ldots, x_m\}$ to represent the state space of X. In our setting, a conditional probability distribution $\Pr(X_i \mid \Pi_{X_i})$ is also denoted a conditional probability table (CPT).

A BN is provided input or *evidence* by clamping zero or more of its nodes to their observed states. Given evidence, answers to different probabilistic queries can be computed by means of a BN. In this context, the concept of an instantiation or explanation over all non-clamped nodes is key, and is formally defined as follows.

Definition 5 (Explanation) *Consider a BN $\beta = (\boldsymbol{X}, \boldsymbol{E}, \boldsymbol{P})$ with $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ and evidence $\boldsymbol{e} = \{X_1 = x_1, \ldots, X_m = x_m\}$ where $0 \leq m < n$. An explanation $\boldsymbol{x}$ is defined as $\boldsymbol{x} = \{x_{m+1}, \ldots, x_n\} = \{X_{m+1} = x_{m+1}, \ldots, X_n = x_n\}$.*

One is often interested in computing answers to queries of the form $\Pr(\boldsymbol{x} \mid \boldsymbol{e})$, and in particular in finding a most probable explanation (MPE). An MPE is an explanation $\boldsymbol{x}^*$ such that $\Pr(\boldsymbol{x}^* \mid \boldsymbol{e}) \geq \Pr(\boldsymbol{x} \mid \boldsymbol{e})$ for any other explanation $\boldsymbol{x}$. In addition to MPE, the computation of posterior marginals, or $\Pr(X \mid \boldsymbol{e})$ for $X \in \boldsymbol{X}$, is of great interest. To compute answers to these queries, both complete and incomplete algorithms for Bayesian network computation may be used. Complete algorithms include clique tree propagation [33, 2, 25, 62], conditioning [55, 11], variable elimination [34, 71, 14], and arithmetic circuit evaluation [12, 9, 8]. Incomplete algorithms, and in particular stochastic local search algorithms, have also been used for MPE [27, 39, 21, 42, 44, 46] as well as MAP [52, 53] computation.

An important distinction exists between algorithms that rely on an off-line compilation step — for example clique tree propagation and arithmetic circuit evaluation — and those that do not — for example variable elimination and stochastic local search. Compilation has several benefits when it comes to integration into resource-bounded systems including hard real-time systems [48, 40]. Our main emphasis in this article is on compilation and in particular the HUGIN clique tree clustering approach [33, 25], in which clique trees play a central role.

Definition 6 (Clique tree) *A clique tree $\beta''' = (\Gamma,\Phi)$ for a BN $\beta = (\boldsymbol{X}, \boldsymbol{E}, \boldsymbol{P})$ consists of nodes (or cliques) Γ and edges Φ. Here, (Γ,Φ) is an undirected graph that is a tree, and needs to adhere to the following conditions: (i) for each clique $\gamma \in \Gamma$, $\gamma \subseteq \boldsymbol{X}$; (ii) each family in $(\boldsymbol{X}, \boldsymbol{E})$ must appear in some clique $\gamma \in \Gamma$; and (iii) if a node $X \in \boldsymbol{X}$ is such that $X \in \gamma_i$ and $X \in \gamma_j$, then $X \in \gamma_k$ for any $\gamma_k \in \Gamma$, where γ_k is on the path between γ_i and γ_j in β'''.*

The HUGIN approach is interesting in its own right, and in addition there is a well-established relationship to arithmetic circuits [54]. A clique tree[2] β''', which is used for on-line computation, is constructed from a BN $\beta = (\boldsymbol{X}, \boldsymbol{E}, \boldsymbol{P})$ in the following way by the HUGIN algorithm [33, 2]: A *moral graph* β' is first constructed by making an undirected copy of β and then augmenting it with moral edges as follows. For each node $X \in \boldsymbol{X}$, HUGIN adds, in its moralization step, to β' a *moral edge* between each pair of nodes in Π_X if no such edge already exists in β'. Second, HUGIN creates a triangulated graph β'' by heuristically adding fill-in edges to β' such that no chordless cycle of length greater than three exists. Third, a clique tree β''' is created from the triangulated graph β''. A clique tree is constructed sequentially, such that for any two nodes γ_i and γ_j in the clique tree, all nodes between them contain $\gamma_i \cap \gamma_j$. This is known as the running intersection property, which (informally) enforces global consistency through local consistency, thereby enabling the computation of marginals and MPEs. Each CPT $\Pr(X \mid \Pi_X) \in \boldsymbol{P}$ is assigned to a clique containing $\{X\} \cup \Pi_X$.

Using β''', HUGIN can compute marginals [33] or MPEs [13]. These computations rely on the following well-known result.

Theorem 7 *Let $\Pr(\boldsymbol{X})$ be a probability distribution induced by a BN β, and let $\beta''' = (\Gamma,\Phi)$ be a corresponding clique tree. Then the distribution can be expressed as:*

$$\Pr(\boldsymbol{X}) = \prod_{\gamma_i \in \Gamma} \Pr(\gamma_i) \Big/ \prod_{\{\gamma_i,\gamma_j\} \in \Phi} \Pr(\gamma_i \cap \gamma_j).$$

The size of a clique tree β''' essentially determines the compilation and propagation times, be it for computation of MPEs or marginals. The following parameters are useful in characterizing clique trees, and thereby also computation times for algorithms that use clique trees.

Definition 8 (Clique tree parameters) *Let $\Gamma = \{\gamma_1, \ldots, \gamma_\eta\}$ be the set of cliques in a clique tree β''' = (Γ, Φ). The largest clique in Γ (in terms of number of BN nodes) is defined as*

$$\ell(\Gamma) = \arg\max_{\gamma \in \Gamma} (|\gamma|), \tag{2}$$

with the cardinality of the largest clique (in terms of number of nodes) is defined as $c(\Gamma) = |\ell(\Gamma)|$. The state space size s of a clique $\gamma \in \Gamma$ is defined as

$$s(\gamma) = |\Omega_\gamma| = \prod_{X \in \gamma} |\Omega_X|, \tag{3}$$

where $X \in \boldsymbol{X}$ is a node in $\beta = (\boldsymbol{X}, \boldsymbol{E}, \boldsymbol{P})$. The maximal clique in Γ (in terms of state space size) is defined as

$$m(\Gamma) = \arg\max_{\gamma \in \Gamma} (|\Omega_\gamma|), \tag{4}$$

[2]For simplicity, and even though they are used to denote slightly different concepts by some authors, we generally do not distinguish between junction trees and clique trees in this article.

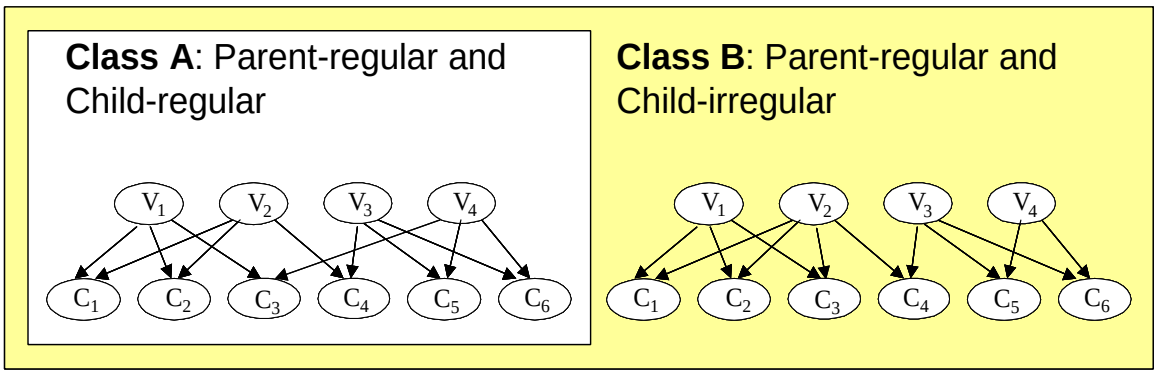

Figure 1: Two classes, Class A and Class B, of bipartite graphs and Bayesian networks (BNs). In both Class A and Class B BNs, all leaf nodes have the same number of parents P (here, $P = 2$). In Class A BNs, all root nodes have k or $k + 1$ children, with $k \geq 0$. In Class B BNs, the number of children may vary between the root nodes.

with maximal clique size $m^*(\Gamma) = s(m(\Gamma))$. *The total clique tree size of* β''' *is defined as*

$$k(\Gamma) = \sum_{\gamma \in \Gamma} s(\gamma) = \sum_{\gamma \in \Gamma} |\Omega_\gamma| \,. \tag{5}$$

The width $w(\Gamma)$ of a clique tree Γ is defined as $w(\Gamma) = c(\Gamma) - 1$, and thus size of a clique tree's largest clique and width are closely related.

In general, there is more than one clique tree for a BN, and it is interesting to consider optimal clique trees, which may be defined as follows.

Definition 9 (Clique tree optimization) *Let* $\mathbf{\Gamma} = \{\Gamma_1, \Gamma_2, \ldots\}$ *be the set of all (clique tree) cliques for clique trees* $\{\beta'''_1, \beta'''_2, \ldots\}$ *for a BN* β. *The clique tree with the optimal (minimal) largest clique (in terms of number of nodes) is defined, using (2), as*

$$L(\mathbf{\Gamma}) = \arg\min_{\Gamma \in \mathbf{\Gamma}} \left(\ell(\Gamma)\right), \tag{6}$$

with optimal largest clique size $\ell^*(\mathbf{\Gamma}) = |\ell(L(\mathbf{\Gamma}))|$. *The optimal clique tree (in terms of minimizing total size) is defined, using (5), as*

$$K(\mathbf{\Gamma}) = \arg\min_{\Gamma \in \mathbf{\Gamma}} \left(k(\Gamma)\right), \tag{7}$$

and with optimal (minimal) total clique tree size defined as $k^*(\mathbf{\Gamma}) = k(K(\mathbf{\Gamma}))$.

It should be noted that some of the quantities, and specifically (2) and (6) in Definition 9 and Definition 8 are strictly graph-theoretic. Other quantities, specifically (7) and (4), also take into account the size of the state spaces of BN nodes, and are consequently of particular interest to researchers interested in the scalability of computation using clique tree clustering.

The performance of many complete BN inference algorithms has been found to depend on treewidth w^*, which is defined as $w^*(\mathbf{\Gamma}) = \ell^*(\mathbf{\Gamma}) - 1$ [33, 15]. When the BN β and its clique trees $\{(\Gamma_1,\Phi_1), (\Gamma_2,\Phi_2), \ldots\}$ are obvious from the context, we simply write $w^* = \ell^* - 1$, where ℓ^* is the minimal largest clique across all clique trees for a BN. Treewidth computation is NP-complete [3], and greedy triangulation heuristics that compute upper bounds on treewidth (or optimal maximal clique size) are typically used in practice [31].

A key research question, which we investigate in this article, is how clique tree size relates to parameters that can be computed for a BN in polynomial time, such as the following parameters:

- $V = |\boldsymbol{V}|$, the number of root nodes in a BN, with $V \geq 1$.
- $T = |\boldsymbol{T}|$, the number of trunk nodes in a BN, with $T \geq 0$.
- $C = |\boldsymbol{C}|$, the number of leaf nodes in a BN, with $C \geq 0$, so the total number of BN nodes is $n = C + V + T$.
- P_{avg}, the average number of parents for all non-root nodes $\bar{\boldsymbol{V}} \subset \boldsymbol{X}$ in a BN, with $1 \leq P_{\text{avg}} \leq N - 1$.
- S_{avg}, the average number of states for all BN nodes $\boldsymbol{X}$, with $S_{\text{avg}} \geq 1$.

Using the parameters above, we study bipartite BNs in detail in this article. In a bipartite BN $\beta = (\boldsymbol{X}, \boldsymbol{E}, \boldsymbol{P})$, the nodes in $\boldsymbol{X}$ are partitioned into root nodes $\boldsymbol{V}$ and leaf nodes $\boldsymbol{C}$ according to the following definition.

Definition 10 (Bipartite DAG) *Let $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ be a DAG. If $\boldsymbol{X}$ can be split into partite sets $\boldsymbol{V} = \{X \in \boldsymbol{X} \mid i(X) = 0\}$ (the root nodes) and $\boldsymbol{C} = \{X \in \boldsymbol{X} \mid i(X) > 0\}$ (the leaf nodes) such that any $(V, C) \in \boldsymbol{E}$ is such that $V \in \boldsymbol{V}$ and $C \in \boldsymbol{C}$, then $\boldsymbol{G}$ is a bipartite DAG.*

While our approach is general, as discussed in Section 3.3, we also note that important classes of application BNs are bipartite or have significant induced subgraphs that are bipartite. Naïve Bayes classifiers are, for example, a special case of bipartite BNs with only one root node. Application areas where bipartite BNs can be found include gas path diagnosis for turbofan jet engines [60], sensor validation and diagnosis of rocket engines [6], diagnosis in computer networks [59], medical diagnosis [64], and decoding of error-correcting codes [37]. A well-known bipartite BN for medical diagnosis is QMR-DT; in it diseases are root nodes and symptoms are leaf nodes [64]. QMR-DT may be used to compute the most likely instantiation of the disease nodes (i.e., the most probable explanation), given known symptoms [64, 24, 50]. In research on decoding of error-correcting codes, a close relationship has been established to Bayesian network computation [38, 37]. It turns out that the subgraph induced by nodes corresponding to the hidden information and codeword bits in a decoding BN forms a bipartite BN [38, Figure 7].

Bipartite BNs also generalize satisfiability (SAT) instances: root nodes correspond to propositional logic variables and leaf nodes correspond to propositional logic clauses [63, 61, 45]. Special inference algorithms have been designed for bipartite BNs; see for example the study of approximate inference algorithms for bipartite BNs by Ng and Jordan [50]. Finally, general BNs often have non-trivial bipartite components, and bipartite BNs therefore form a stepping stone for these more general, multi-partite BNs.

For the purpose of this article, our emphasis is on randomly generated BNs, as this approach admits a very systematic investigation of BN inference algorithms [66, 23, 45]. Bipartite BNs are generated randomly using the BPART algorithm [45], which is a generalization of an algorithm that randomly generates hard and easy problem instances for satisfiability [47]. The BPART algorithm, for which we use the signature BPART(V, C, P, S, R), operates as follows.[3] First, $V = |\boldsymbol{V}|$ root nodes and $C = |\boldsymbol{C}|$ leaf nodes, all with S states, are created. The value of the binary input parameter R determines whether regular Class A (R = **true**) or irregular Class B (R = **false**) BNs are generated (see Figure 1). In Class B BNs, P parent nodes $\{X_1, \ldots, X_P\}$ are, for each leaf node, picked uniformly at random without replacement among the V root nodes. In Class A BNs, which form a strict subset of Class B BNs [45], parents are picked such that all root nodes have exactly k or $k + 1$ children for some $k \geq 0$. Conditional probability tables (CPTs) of all nodes are also constructed by BPART; however in this article we focus on the impact of the structural parameters V, C, $P = P_{\text{avg}}$, and $S = S_{\text{avg}}$ on clique tree size. As defaults, parameter values $S = 2$ and $R =$ **false** are employed, and we use BPART(V, C, P) as an abbreviation for BPART(V, C, P, 2, **false**). An additional default of $P = 2$ is also used, giving BPART(V, C) as an abbreviation for BPART(V, C, 2). Since, in a given context we always fix P, the total number of edges in a BPART(V, C, P) BN is clearly $E = C \times P$.

Here is an example of using clique tree clustering on a small BPART BN.

Example 11 (BPART BN) *Figure 2 shows how a* BPART *BN may be compiled into a clique tree. For each BN leaf node $C \in \{C_1, C_2, C_3, C_4, C_5, C_6\}$, a clique is created. In addition, there are two cliques containing BN root nodes only, namely the cliques $\{V_1, V_2, V_4\}$ and $\{V_2, V_3, V_4\}$.*

Note that clique tree clustering's moralization step, which creates a moral graph β' from a BPART BN β, ensures that there are edges between all P root nodes that share a leaf node. To keep the discussion succinct, we often say that BPART creates moral edges without explicitly mentioning the moralization step, even though moralization actually creates the edges when a BPART BN is compiled. The compilation of the bipartite BN in Figure 2 illustrates the crucial formation of cycles in a BN's moral graph and the resulting generation of fill-in edges.

In the bipartite case, all non-root nodes are leaf nodes and $n = C + V$. It has been shown analytically and empirically that the ratio of C to V, the C/V-ratio, is a key parameter for BN inference hardness [45]. (We consider only non-empty BNs and so $V \geqslant 1$ and the C/V-ratio is always well-defined.) Specifically, the C/V-ratio can be used to predict upper and lower bounds on the optimal maximal clique size (or treewidth) of the induced clique tree for BNs randomly generated using the BPART algorithm. Taking this approach, upper bounds on optimal maximal clique sizes as well as inference times have been computed. Using regression analysis, the mean number of nodes in the maximal clique was found to be approximately linear in the C/V-ratio. This linear growth translates into an

[3] The more extensive signature BPART(Q, F, V, C, S, R, P) was previously used [45]. Here, the Q and F parameters are used to control the conditional probability table (CPT) types of BN root and non-root nodes respectively. The parameter R is used to control the regularity in the number of children of root nodes. Since our emphasis in this article is on the impact of the parameters V, C, S, and P, we typically use the default values for Q, F, and R , and generally simplify the signature to BPART(V, C, P, S).

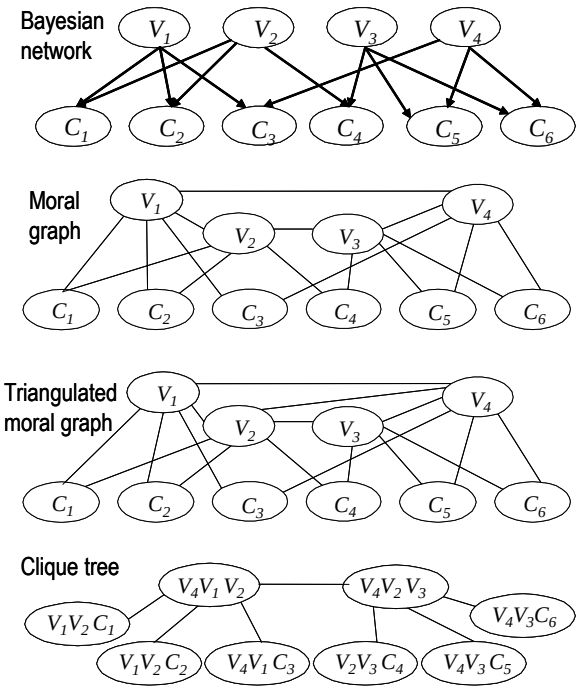

Figure 2: Compilation of BPART BN β (top) to clique tree β''' (bottom). There is a loop (V_1, V_2, V_3, V_4) in the moral graph β', leading to a fill-in edge (V_2, V_4) in triangulated graph β'', which again leads to cliques $\{V_4, V_1, V_2\}$ and $\{V_4, V_2, V_3\}$ in the clique tree β'''.

approximately exponential growth in maximal clique size — and consequently in clique tree clustering computation time — as a function of the C/V-ratio. This has been found to be true for both Class A and Class B BPART BNs [45].

Because w^*, ℓ^*, and k^* are hard to compute, one often computes upper bounds $\bar{w}^*$, $\bar{\ell}^*$, and $\bar{k}^*$ using various heuristics. We now focus on three upper bounds $\bar{k}^*_T$, $\bar{k}^*_E$, and $\bar{k}^*_S$ for k^*. The trivial upper bound $\bar{k}^*_T = n \times d^{(w^*+1)}$, where $Y = \arg\max_{X \in \boldsymbol{X}} (|\Omega_X|)$ and $d = |\Omega_Y|$, is well-known; an easy extension[4] is $\bar{k}^*_E = (n - \bar{w}^*) \times d^{(\bar{w}^*+1)}$. Another upper bound is obtained by summing the sizes of cliques, or $\bar{k}^*_S(\beta) = k(\Gamma)$, where Γ is computed from β using HUGIN [28]. A few observations can be made in regard to these and related upper bounds. First, they may assume that the treewidth w^* is known. But it is NP-hard to compute treewidth w^* [3], and therefore bounds involving treewidth are not directly useful. Consequently, we use an upper bound $\bar{w}^* \geq w^*$ when forming an upper bound on $\bar{k}^* \geq k^*$. Such upper bounds $\bar{w}^*$ can be computed using heuristics algorithms such as HUGIN.

To empirically investigate these upper bounds, we consider a few existing BNs, generated using the signature BPART(V, C, P) = BPART(30, 60, 3) [45, Table 6]. With reference to the above bounds $\bar{k}^*_T$ and $\bar{k}^*_E$, we thus have $n = 30 + 60 = 90$ and $d = S = 2$. Here, $\bar{w}^*$ as well as $\bar{k}^*_S$ are computed by an implementation of the HUGIN algorithm. Results for these BNs are presented in Table 1, which shows that the bounds $\bar{k}^*_T$ and $\bar{k}^*_E$ are not very good compared to $\bar{k}^*_S$. Specifically, we see in the relative size columns of Table 1 the following. The size of the computed total clique tree size $\bar{k}^*_S$ relative to the trivial upper bound $\bar{k}^*_T$ ranges from 1.20% (worst case) to 6.72% (best case), and the easy upper bound $\bar{k}^*_E$ is only slightly better. Previously, a difference of an order of magnitude or more between the two bounds $\bar{k}^*_T$ and $\bar{k}^*_S$ has been found across a broad range of benchmark instances [51]. These empirical results illustrate why it is useful to consider $\bar{k}^*_S$, as computed by clique tree clustering algorithms, instead of more obvious bounds such as $\bar{k}^*_T$ and $\bar{k}^*_E$, and our growth curves in the rest of this article are based on $\bar{k}^*_S$.

In larger BNs, it is important but also very difficult to understand and predict clique tree clustering's cycle-generation and fill-in processes, which again determine maximal clique size and total clique tree size. A main contribution of this article, further discussed in Section 3, Section 4, and Section 5, is how we improve the understanding of the growth of total clique tree size $\bar{k}^*_S$ as a function of BPART BN growth. Whether one computes MPEs or marginals, the structure and size of the clique tree is the same, and only the numerical operations (maximization for MPE computation $\Pr(\boldsymbol{x}^* \mid \boldsymbol{e})$ versus addition for marginal computation $\Pr(X \mid \boldsymbol{e})$ for $X \in \boldsymbol{X}$) differ. The clique tree growth curves discussed in this article apply to both cases.

[4]The extension was introduced by an anonymous reviewer.

BN	Maximal Clique Bound $2^{\bar{\ell}^*}$	Tree-width Bound $\bar{w}^*$	Clique Tree Size Bound - Trivial $\bar{k}_T^*$	Clique Tree Size Bound - Easy $\bar{k}_E^*$	Clique Tree Size Bound - Sum $\bar{k}_S^*$	Relative Size - $100\bar{k}_S^*/\bar{k}_T^*$	Relative Size - $100\bar{k}_S^*/\bar{k}_E^*$
β_0	16,384	13	1,474,560	1,261,568	61,056	4.14%	4.84%
β_{12}	16,384	13	1,474,560	1,261,568	75,840	5.14%	6.01%
β_{73}	16,384	13	1,474,560	1,261,568	81,200	5.51%	6.44%
β_{89}	16,384	13	1,474,560	1,261,568	99,138	6.72%	7.86%
β_{92}	16,384	13	1,474,560	1,261,568	53,344	3.62%	4.23%
β_6	262,144	17	23,592,960	19,136,512	439,776	1.86%	2.30%
β_{80}	262,144	17	23,592,960	19,136,512	284,192	1.20%	1.49%

Table 1: Three different upper bounds on total clique tree size for seven example bipartite Bayesian networks.

3 Developing Model-Based Reasoners using Bayesian Networks

The development of model-based reasoners, including those that use Bayesian networks, typically involves an iterative or spiral process. One starts with a simple model, which is refined and extended as further information, experimental results, or additional requirements become available. In other words, an iterative development process often manifests itself as model growth, for example Bayesian network growth. More specifically, if we consider bipartite Bayesian networks used for diagnosis [64, 6, 59, 60], we may identify at least two forms of growth:

- Growth in the number of root nodes V, to capture additional faults that may occur in the system being modeled. In a gas path diagnosis BN, these root nodes represent health parameters for a turbofan engine [60], and by increasing the number of health parameters a more comprehensive diagnosis can be computed. In a BN for medical diagnosis, additional root nodes may be introduced because one wants to consider more diseases [64].

- Growth in the number of leaf nodes C, to represent additional evidence that can be used to distinguish between the underlying faults by computing marginals, MPE, or MAP. In a BN for gas path diagnosis, these leaf nodes can represent additional measurements made on the turbofan engine [60]. In a BN for medical diagnosis, these leaf nodes may represent additional symptoms or tests [64].

A hypothetical BN development process, where small BNs are used for the purpose of illustration, is provided in Figure 3. The figure shows two different BN growth paths leading from a BN with $V = 2$ and $C = 4$ (lower left corner of Figure 3) to a larger BN with $V = 4$ and $C = 6$ (upper right corner of Figure 3).

Even though we place emphasis on growth or increase here, it is really the concept of change that is important. Our results apply to change in general, both increases and decreases in BN size, however the increase or growth perspective is more prevalent. For example, both in knowledge engineering and BN structure learning one typically proceeds by growing a BN iteratively. In addition, we want to emphasize the connection between BN growth and biological and medical growth processes [4, 35, 17] as well as growth of random graphs (see Section 4). Similar to these areas of research, this article represents a shift away from a particular BN β to families or sequences $(\beta(1), \beta(2), \beta(3), \beta(4), \dots)$ of BNs and the processes by which BNs are developed or grown. BN growth processes might be automatic, as in machine learning or data mining; manual, as in knowledge engineering by direct manipulation of a BN; or semi-automatic, as when editing a high-level language from which BNs are auto-generated [43].

An illustration of the connection between BN growth and clique tree growth is provided in Figure 4. This figure illustrates why it is important to vary a cause (say, the number of leaf nodes in BNs or the density of edges in the moral graphs of BNs) such that a wide range of effects (different clique tree sizes) can be studied. At the highest level, we want to communicate two main ideas in this article. The first idea is the use of a macroscopic growth curve $g_T(x)$ for total clique tree size, where x is an independent parameter. As an illustration, $g_T(x)$ for bipartite BNs is emphasized in Section 3.2, but the approach clearly generalizes beyond bipartite BNs as discussed in Section 3.3. As a second idea, discussed in Section 3.1, we investigate different independent parameters x in $g_T(x)$. The use of $x = C/V$, where C is number of leaf nodes and V is number of root nodes, is well-known. A novel aspect of this work is the investigation of an alternative to C/V.

The research on the BPART algorithm and its generalization, the MPART algorithm, extends existing research on generating hard instances for the satisfiability problem [47] as well as existing research on randomly generating BNs [66, 5, 27, 11, 52, 22, 23, 45]. Our work on BPART in this article is different from previous research [45] in several ways including the following: The emphasis in this article is on total clique tree size instead of size of largest clique, and in particular we form total clique tree size by carefully partitioning the cliques $\mathbf{\Gamma}$ in the clique tree.

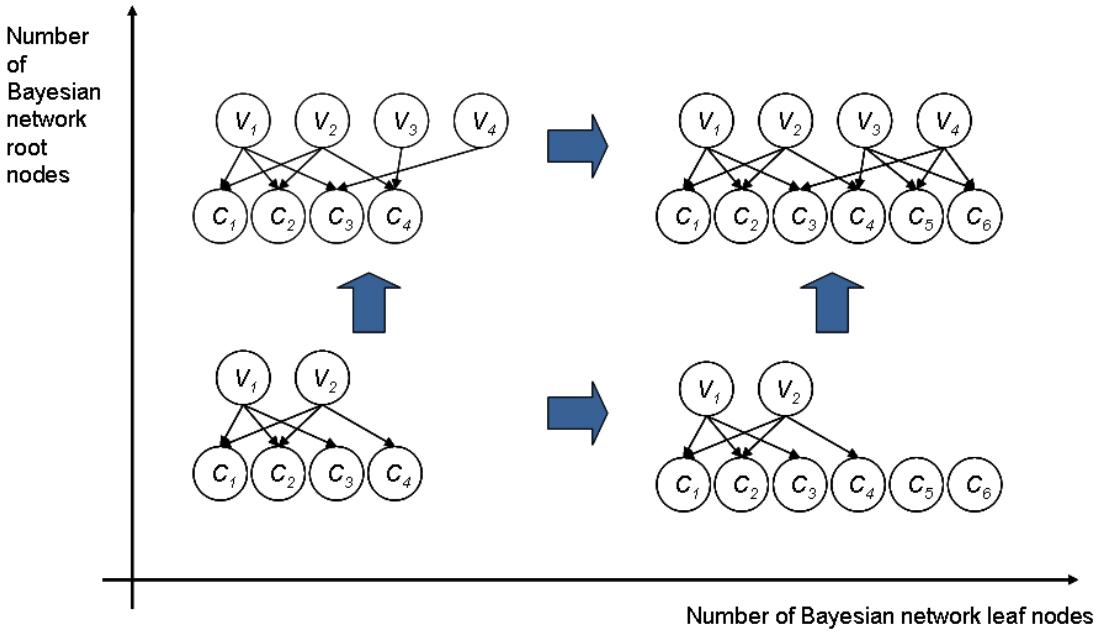

Figure 3: An example of how a bipartite BN with $V = 4$ root nodes and $C = 6$ leaf nodes (top right) may be developed or grown from a bipartite BN with $V = 2$ root nodes and $C = 4$ leaf nodes (bottom left).

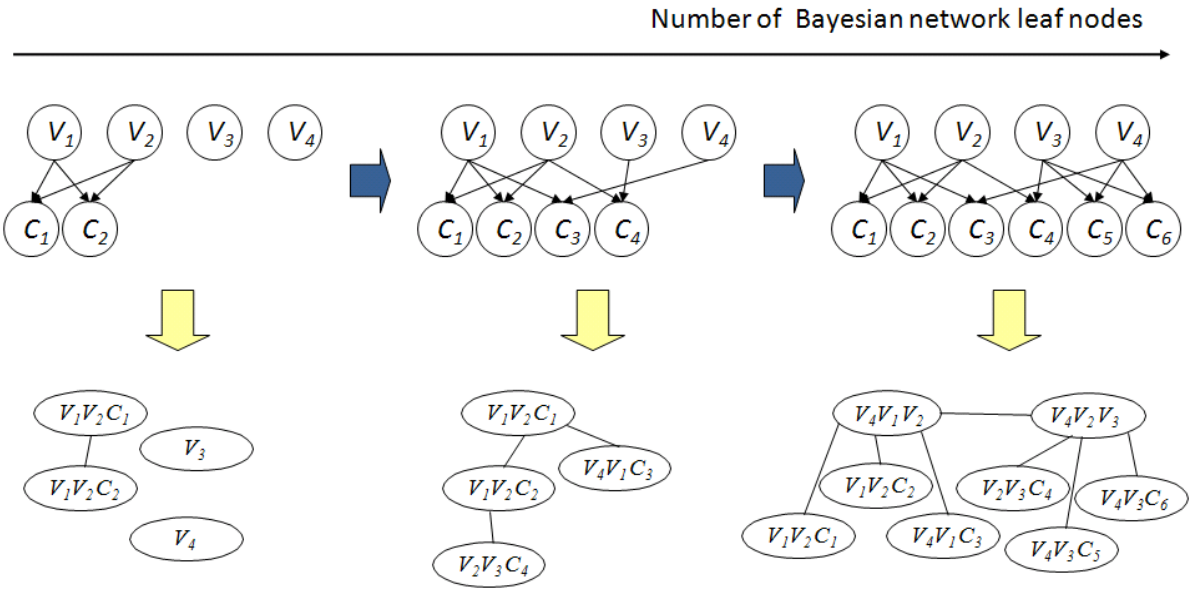

Figure 4: How clique tree size (bottom) varies when the number of BN leaf nodes is varied (top). Horisontally, this figure illustrates how a BN may grow by having leaf nodes added. Vertically, this figure shows how BNs are compiled into clique trees. The growth of a BPART(4, 2) BN (top left) into a BPART(4, 4) BN (top middle) and finally a BPART(4, 6) BN (top right) is depicted in the top row. Clique trees compiled from these BNs are shown in the bottom row.

3.1 Independent Parameters for Bayesian Networks and Moral Graphs

Let W be the random number of moral edges in the moral graph β' of a randomly generated BN β. Then $E(W)$ is the expected number of moral edges. It turns out to be fruitful to use $x = E(W)$ as the independent parameter in the growth curve $g_T(x)$. In the rest of this section we discuss this issue in more detail.

3.1.1 Balls and Bins

The balls and bins model, where balls are placed uniformly at random into bins, turns out to be useful in our analysis of clique tree clustering's moralization step. Following the balls and bins model, we let m denote the number of balls and n denote the number of bins. Further, we let X and Y be random variables representing the number of empty and occupied bins respectively. The expected number of empty bins X is

$$E(X) = n\,(1 - 1/n)^m\,. \tag{8}$$

The expected number of occupied bins Y is

$$E(Y) = n\,(1 - (1 - 1/n)^m)\,. \tag{9}$$

It is well-established that the expected number of empty bins X can be approximated as

$$E(X) \approx ne^{-m/n}, \tag{10}$$

while the expected number of occupied bins Y is approximated by

$$E(Y) \approx n\left(1 - e^{-m/n}\right). \tag{11}$$

How does the balls and bins model apply to the moral graph created, using clique tree clustering, from a bipartite BN? We restrict[5] our attention to the subgraph of β' induced by $\boldsymbol{V}$, abbreviated $\beta'[\boldsymbol{V}]$. The bins are all possible edges in the moral subgraph $\beta'[\boldsymbol{V}]$, and BN leaf nodes induce actual edges (corresponding to balls) in the moral graph. For clarity, we say edge-bin instead of bin and edge-ball instead of ball. The formal definition that makes the connection between the balls and bins model and a bipartite Bayesian network is as follows.

Definition 12 (Edge-balls and edge-bins) *Consider the root nodes $\boldsymbol{V} \subseteq \boldsymbol{X}$ in a bipartite BN $\beta = (\boldsymbol{X}, \boldsymbol{E}, \boldsymbol{P})$. An* edge-bin *is a possible edge in the BN's moral subgraph $\beta'[\boldsymbol{V}]$, namely an edge between two root nodes $\{V_i, V_j\}$, where $V_i, V_j \in \boldsymbol{V}$ and $i \neq j$. The set of all* edge-bins *is $\{\{V_i, V_j\} \mid V_i, V_j \in \boldsymbol{V}$ and $i \neq j\}$, and an edge-ball is placed into an edge-bin by picking one edge-bin uniformly at random with probability $p = 1 \Big/ \binom{|\boldsymbol{V}|}{2}$.*

We use, as will be seen shortly, a balls and bins approach to obtain the expected number of moral edges in the moral graphs induced by distributions of BNs, specifically BPART(V, C, P) BNs. We now consider bipartite BNs where leaf nodes have exactly two parents (Section 3.1.2) or an arbitrary number of parents (Section 3.1.3).

3.1.2 Balls and Bins: Two Parents

In the BPART(V, C, 2) model, all edge-bins are uniformly and repeatedly eligible for placing edge-balls into. In other words, we have sampling with replacement. Here is an example of applying our balls and bins model in the BPART setting.

Example 13 *Figure 5 shows a BN sampled from the* BPART*(V, C, P) distribution with $V = 4$, $C = 6$, and $P = 2$. For this particular BN, 4 of the 6 possible edge-bins contain edge-balls as can be seen in the subgraph induced by the root nodes $\{V_1, V_2, V_3, V_4\}$ to the right in Figure 5.*

Intuitively, as the C/V-ratio increases, it gets more and more likely that a given moral edge gets picked two or more times, or in other words that an edge-bin contains two or more edge-balls. This intuitive argument is formalized in the following result, where we shorten the expectation $E(W; C, V)$ to $E(W)$ when C and V are obvious.

[5]The reason for this restriction is clarified in Section 3.2, where we discuss growth curves $g_R(x)$ and $g_M(x)$. The growth of total clique tree size due to changes in $\beta'[\boldsymbol{V}]$, for example induced by an increasing $x = C/V$, is captured by $g_R(x)$.

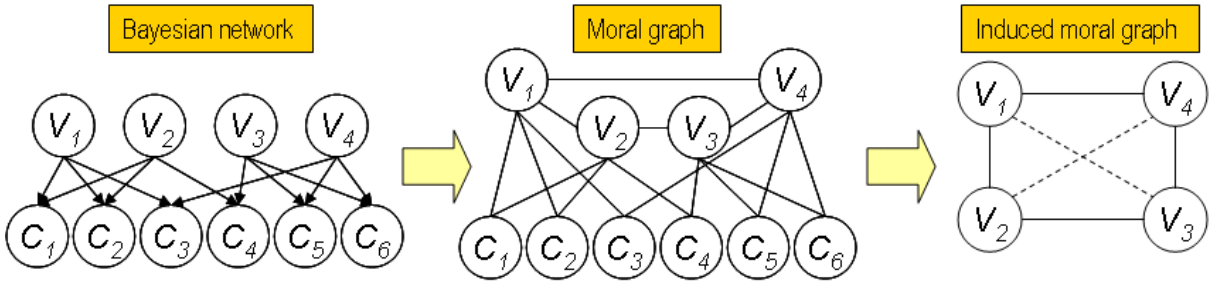

Figure 5: As part of compilation to a clique tree, a bipartite Bayesian network (left) is transformed into a moral graph with moral edges (middle). We focus on the root nodes $\{V_1, V_2, V_3, V_4\}$ and in particular the moral edges in the moral graph's subgraph induced by the root nodes (right). Both the moral edges actually created (edge-bins filled with edge-balls as shown using solid lines) as well as the potential moral edges not created (edge-bins *not* filled with edge-balls as shown using dashed lines) are shown to the right above.

Theorem 14 (Moral edges, exact) *Let the number of moral edges created using* BPART*(V, C, P) be a random variable W. The expected number of moral edges $E(W; C, V)$ is, for $P = 2$, given by:*

$$E(W; C, V) = \binom{V}{2}\left(1 - \left(1 - 1 \Big/ \binom{V}{2}\right)^{C}\right). \tag{12}$$

Proof. We use the balls and bins model. Here, the edge-balls correspond to leaf nodes, of which there are $m = C$. The edge-bins are all possible moral edges, of which there are $n = \binom{V}{2}$ in a bipartite graph with V root nodes. Plugging m and n into (9) gives the desired result (12). ■

It is sometimes convenient to use the following approximation for $E(W; C, V)$ in (12).

Theorem 15 (Moral edges, approximate) *Let the number of moral edges created using* BPART*(V, C, P) be a random variable W. The expected number of moral edges $E(W; C, V)$ is, for $P = 2$, approximated as follows:*

$$E(W; C, V) \approx \binom{V}{2}\left(1 - \exp\left(-C \Big/ \binom{V}{2}\right)\right). \tag{13}$$

Proof. We use the balls and bins model. Here, the edge-balls correspond to leaf nodes, of which there are $m = C$. The edge-bins are all possible moral edges, of which there are $n = \binom{V}{2}$ in a bipartite graph with V root nodes. Plugging m and n into (11) gives the desired result (13). ■

Given (12) and (13), we can make a few remarks. In contrast to the C/V-ratio or the E/V-ratio, the expectation $E(W)$ takes into account the effect of picking parents among pairs of BN root nodes with replacement. For low values of C/V or E/V one would not expect the effect of replacement to be great, but for large C/V- or E/V-ratios the difference may be substantial as illustrated in the following examples.

Example 16 ($C = 30$ leaf nodes) *Let $V = 30$, $C = 30$, and $P = 2$. The expected number of moral edges is $E(W) = 28.99$ using (12) and $E(W) \approx 29.02$ using (13).*

Example 17 ($C = 300$ leaf nodes) *Let $V = 30$, $C = 300$, and $P = 2$. The expected number of moral edges is $E(W) = 216.91$ using (12) and $E(W) \approx 216.74$ using (13).*

In Example 16, where $E(W) \approx C$, it is relatively unlikely that there are edge-bins with two or more edge-balls. In Example 17, on the other hand, it is very likely that there are edge-bins with two or more edge-balls, and $E(W) < C$. In other words, adding the last leaf node has on average a smaller net effect on the number of moral edges in Example 17 than in Example 16, and this is captured in $E(W)$ but not in C/V or E/V. This is important because the essential difference, as far as cycle (and thus clique) formation in clique tree clustering is concerned, is between (i) no edge-ball and (ii) one or more edge-balls.

3.1.3 Balls and Bins: Arbitrary Number of Parents

We now turn to BPART instances in which P is an arbitrary positive integer. The fundamental complication, as far as the expected number of moral edges $E(W)$ is concerned, is this. For $P > 2$, BPART uses a combination of sampling

with replacement and sampling without replacement: Picking the parents of a given leaf node C_i amounts to sampling without replacement, while picking parents for C_i when parents of C_j are already known (for $i > j$) amounts to sampling with replacement.

We now introduce, for the purpose of approximation, a variant BPART$'$ which works exactly as BPART except that the moral edges induced by the P parent nodes are, for a given leaf node C_i, picked independently and with replacement from the moral graph β'. This means that we in Theorem 18 need to consider sampling with replacement only.

Theorem 18 (Moral edges, exact) *Consider* BPART$'(V, C, P)$ *and let the number of moral edges created be a random variable* Z*. The expected number of moral edges is:*

$$E(Z; C, V, P) = \binom{V}{2}\left(1 - \left(1 - 1 \Big/ \binom{V}{2}\right)^{C\binom{P}{2}}\right). \tag{14}$$

Proof. We use the balls and bins model, and again the number of edge-bins is $n = \binom{V}{2}$ in a bipartite graph with V root nodes. Since BPART$'$ employs sampling with replacement, the number of edge-balls is $m = C \times \binom{P}{2}$. Plugging m and n into (9) gives the desired result (14). ■

We note that Theorem 18 is a generalization of Theorem 14, and abbreviate $E(Z; C, V, P)$ as $E(Z)$ when C, V, and P are clear from the context. Further, we note that $E(Z)$ in (14) can be approximated in a way similar to the approximation of (12) by (13).

We now consider two areas where BPART$'$ works differently than BPART. First, as mentioned above, there is the issue of picking parents with replacement versus without replacement. For BPART(V, C, P), selecting P parents of a leaf node C_i creates exactly $\binom{P}{2}$ edges in the moral graph, since the parents are distinct and $|\Pi_{C_i}| = P$. For BPART$'$(V, C, P), on the other hand, we end up with $\binom{P}{2}$ edge-balls placed into edge-bins, and consequently at most $\binom{P}{2}$ edges in the moral graph. A second issue is how edge-balls are placed by BPART$'$; specifically, the edge-bins picked might not form a clique. In summary, we note that $E(Z)$ is an approximation for $E(W)$ for BPART(V, C, P) for $P > 2$, justified in part by the well-known fact that sampling without replacement can be approximated using sampling with replacement as the number of objects sampled from (here, the V root nodes) grows.

Why are the above balls and bins models of BN moralization interesting? The reason is that we are concerned with the possible *causes*, at a macroscopic level, that influence clique tree size. The expected number of moral edges is one such cause or independent parameter x. In the context of random BNs generated by BPART, we indirectly control the placement of moral edges, since we place constraints on the structure of these BNs through BPART's input parameters. When it comes to the *effect*, namely tree clustering performance, it is natural to optimize (minimize) the size of the maximal clique. Since this is hard [3], current algorithms including HUGIN use heuristics that upper bound optimal maximal clique size ℓ^* and clique tree size k^* using $\bar{\ell}^*$ and $\bar{k}^*$ respectively. Such upper bounds on clique tree size are just referred to as clique tree sizes in the following, and we seek in Section 3.2 a closed form expression $y = g(x)$ for the dependent parameter clique tree size as a function of the independent parameter x.

3.2 Growth Curves for Bipartite Bayesian Networks

Here, we develop models of restricted clique tree growth that extend exponential growth curves [45] used to model unrestricted growth. Even though Bayesian networks and clique trees are discrete structures, we approximate their growth by using continuous growth curves (or growth functions) in order to simplify analysis. We discuss bipartite BNs in this section and generalize to arbitrary BNs in Section 3.3.

For bipartite BNs, including BPART BNs, there are two types of nodes in the clique tree as reflected in the following definition.

Definition 19 (Root clique, mixed clique) *Consider a clique tree* β''' *with cliques* Γ *constructed from a bipartite BN* β*. A clique* $\gamma \in \Gamma$ *is denoted a* root clique *if all the BN nodes in* γ *are root nodes in* β*. A clique* $\gamma \in \Gamma$ *is denoted a* mixed clique *if* γ *contains at least one root node and at least one leaf node.*

It is easy to see that root and mixed cliques are the only clique types induced by bipartite BNs, and that root cliques are interior nodes in the clique tree while mixed cliques are leaf nodes. An illustration of Definition 19 is provided by the clique tree in Figure 6. The BN from which this clique tree is compiled is depicted in Figure 2 and in Figure 4.

We now consider clique trees generated from random BNs. Random variables K_T, K_R, and K_M are used to represent the total clique tree size, the size of all root cliques, and the size of all mixed cliques respectively:

$$K_T = K_R + K_M. \tag{15}$$

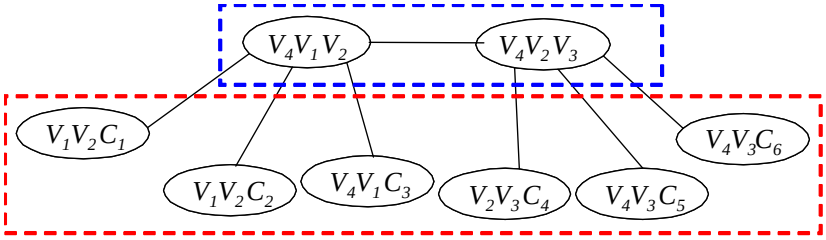

Figure 6: The clique tree for a bipartite Bayesian network, with the two partitions of cliques indicated. Here, $V_4V_1V_2$ and $V_4V_2V_3$ are root cliques (with growth curve g_R) while the remaining six cliques are mixed cliques (with growth curve g_M).

Of particular interest are upper bounds $\bar{k}_S^*$, with corresponding random variables $\bar{K}_S^*$, for which we have $\Pr(\bar{K}_S^* = \bar{k}_S^*)$ and $\bar{K}_S^* = K_T$ in (15). Total clique tree size is the sum of the clique sizes of both types, as is appropriate for clique tree clustering algorithms including HUGIN. We use (15) and linearity of expectation to obtain

$$\begin{aligned} E(K_T) &= E(K_R) + E(K_M) \\ \mu_T &= \mu_R + \mu_M. \end{aligned} \tag{16}$$

In the experimental part of this article, μ_R will be estimated using its sample mean $\hat{\mu}_R$. Collections of such sample means, or the raw data sets themselves, can then be used to construct growth curves by means of regression.

Let X be the predictor (or independent) random variable, and Y the response (or dependent) random variable. In a regression setting, one is interested in the conditional expectation

$$\mu(x) = E(Y \mid X = x) = \int y f(y \mid x) dy,$$

which along with (16) gives $\mu_T(x) = \mu_R(x) + \mu_M(x)$, which are deterministic functions of x. Here, x represents variation in one or more of BPART's input parameters. For instance, C may be varied while V, P, and S are kept constant; see Section 3.1 for details.

The discussion above is intended as a background for understanding the benefit of quantitative growth curves, which we now introduce.

Definition 20 (Clique tree growth curve) *Let $g_R : \mathbb{R} \to \mathbb{R}$ and $g_M : \mathbb{R} \to \mathbb{R}$. Further, let $g_R(x)$ be the growth curve for all root cliques and $g_M(x)$ the growth curve for all mixed cliques. The (total) clique tree growth curve for a bipartite BN is defined as*

$$g_T(x) = g_R(x) + g_M(x).$$

Given Definition 20, we provide a qualitative discussion of the growth of BPART BNs in terms of the C/V-ratio, and put $x = C/V$. This discussion is supported by previous (see [45, 41]) and current (see Section 5) experiments, as well as the connection between BPART and random graphs (see Section 4), and motivates our introduction of growth curves below. In order to keep our discussion relatively simple, we identify three broad stages of clique tree growth, reflecting the growth of root cliques: The initial growth stage, the rapid growth stage, and the saturated growth stage. The *initial growth stage*, where the C/V-ratio is "low", is characterized by "few" leaf nodes relative to the number of root nodes. There is consequently a relatively low contribution by root cliques to the clique tree. In terms of $g_T(x)$, this stage is dominated by mixed cliques and $g_M(x)$. Indeed, as $C/V \to 0$ there are no root cliques with more than one root node. During the *rapid growth stage*, where the C/V-ratio is "medium", root cliques of non-trivial size start emerging, due to formation of cycles where fill-in edges are required in order to triangulate the moral graph. An example of the emergence of such a cycle can be seen in Figure 2 and Figure 4. In this stage, and due to the addition of fill-in edges, the root cliques and $g_R(x)$ gradually overtake the mixed cliques in terms of their contribution to total clique tree size $g_T(x)$. The *saturated growth stage*, where the C/V-ratio is "high", is characterized by a "large" number of leaf nodes relative to the number of root nodes. As C/V approaches infinity, one root clique with V BN root nodes (and size S^V in the BPART model) emerges. In this stage, the mixed cliques eventually start to dominate again, since there is one root clique which has reached its maximal size and cannot grow further. However, since the root clique size is exponential in the number of root nodes, it typically takes a long time before the mixed cliques start dominating again. For large V this effect can be disregarded, and main focus should generally be on the rapid growth stage, and in particular the early part of it, in the transition between the initial and rapid growth stages, where $g_R(x)$ is becoming the dominating factor in $g_T(x)$.

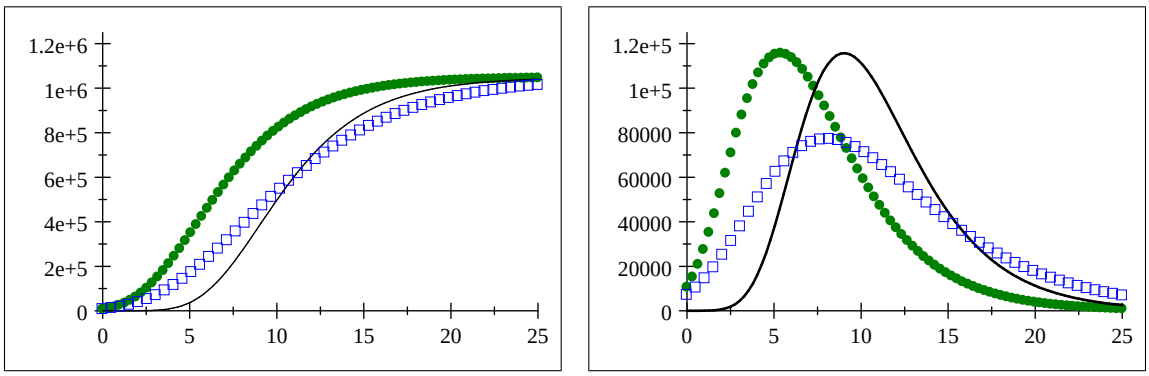

Figure 7: *Left:* Gompertz curves $g_1(x) = 2^{20}e^{-5e^{-0.3x}}$ (green dotted curve), $g_2(x) = 2^{20}e^{-15e^{-0.3x}}$ (black solid curve), and $g_3(x) = 2^{20}e^{-5e^{-0.2x}}$ (blue boxed curve). *Right:* Growth rates $g_1'(x)$, $g_2'(x)$ and $g_3'(x)$ for the Gompertz growth curves.

In the following, we will often discuss $g_R(x)$ and $g_M(x)$ independently. In fact, as reflected in the informal discussion above, the growth curve for mixed cliques $g_M(x)$ is generally less dramatic than the growth curve for root cliques $g_R(x)$. Therefore, we will place more emphasis on $g_R(x)$ in this article, and investigate restricted growth curves suitable for representing this function.

A number of sigmoidal growth curves ("S-curves") have been used to model restricted growth, including the logistic, Gompertz, Complementary Gompertz, and Richards growth curves [4, 35, 17]. For restricted growth curves, $\lim_{x\to\infty} g(x)$ always exists and we define the restricting asymptote as

$$g(\infty) := \lim_{x\to\infty} g(x). \tag{17}$$

For unrestricted growth curves, including the exponential growth curve, $\lim_{x\to\infty} g(x)$ does not exist and there is no asymptote $g(\infty)$ as in (17).

It turns out that restricted Gompertz growth curves give very good approximations of root clique growth $g_R(x)$ for BPART(V, C), see Section 5, and we now study this family of curves in more detail.

Definition 21 (Gompertz growth curve) *Let ζ , $\gamma \in \mathbb{R}$ with $\zeta > 0$ and $\gamma > 0$. A Gompertz growth curve is defined as*

$$g(x) = g(\infty)e^{-\zeta e^{-\gamma x}}. \tag{18}$$

We now discuss some general properties of the form of $g(x)$ in (18). For $x = 0$, clearly $e^{-\gamma x} = 1$, giving $g(0) = g(\infty)e^{-\zeta}$ in (18). In other words, the intersection of $g(x)$ with the y-axis is determined by the parameters $g(\infty)$ and ζ in $g(\infty)e^{-\zeta}$. On the other hand, as $x \to \infty$ in (18), $e^{-\gamma x}$ tends to 0, meaning that $e^{-\zeta e^{-\gamma x}}$ tends to 1 and thus $\lim_{x\to\infty} g(x) = g(\infty)$. The greater γ is, the faster $e^{-\gamma x}$ tends to zero, leading to faster convergence to the asymptote $g(\infty)$.

The derivative $g'(x)$ of the Gompertz growth curve is

$$g'(x) = \frac{d}{dx}g(x) = g(\infty)\zeta\gamma e^{-\gamma x}e^{-\zeta e^{-\gamma x}}, \tag{19}$$

and expresses the *growth rate* of $g(x)$; clearly $g'(x) > 0$ given our assumptions in Definition 21.

In Figure 7 we investigate graphically a few examples of how the parameters $g(\infty)$, ζ, and γ impact the shapes of Gompertz curves. The parameter $g(\infty) = 2^{20}$ is obtained, for example, by considering bipartite BNs with $V = 20$ binary ($S = 2$) root nodes. Figure 7 also shows how the growth rate $g'(x)$ changes when the parameters ζ and γ are varied. Let us first vary ζ as shown in Figure 7. By *increasing* ζ from $\zeta = 5$ to $\zeta = 15$ while keeping $\gamma = 0.3$ constant, the x-location of maximal growth rate $g'(x)$ is *increased* as well. However, the value of $g'(x)$ at its maximum does not change. Let us next vary γ as is also illustrated in Figure 7. As γ *decreases* from $\gamma = 0.3$ to $\gamma = 0.2$, while $\zeta = 5$ is kept constant, the x-location of maximal $g'(x)$ *increases*. In addition, the maximal value of $g'(x)$ decreases with γ decreasing, and generally growth gets more gradual as γ decreases.

In the context of BNs, the independent variable x for the growth curve $g(x)$ may be parametrized using $x = C$, $x = C/V$, $x = E/V = CP/V$, or $x = E(W)$, depending on the data available and the purpose of the model. We now introduce, for BPART, a total growth curve model that includes a Gompertz growth curve.

Theorem 22 (BPART Gompertz growth curve) *The total growth curve $g_T(x)$ for* BPART*(V, C, P, S), assuming Gompertz growth for root cliques and where $x = C$ is the independent variable, is*

$$g_T(x) = S^V e^{-\zeta e^{-\gamma x}} + xS^{P+1}. \tag{20}$$

Proof. Since BPART BNs are bipartite, the growth curve has the form $g_T(x) = g_R(x) + g_M(x)$, where $g_R(x) = g_R(\infty)e^{-\zeta e^{-\gamma x}}$ because we have the Gompertz growth curve. For BPART(V, C, P, S) we have $g_R(\infty) = S^V$, and therefore $g_R(x) = S^V e^{-\zeta e^{-\gamma x}}$ for appropriate choices of ζ and γ. Total mixed clique size is $C \times S^{P+1}$ [45], and hence $g_M(x) = xS^{P+1}$. By forming $g_R(x) + g_M(x)$ we obtain the desired result (20). ■

Analytical growth models or growth curves have been used to model growth of organisms and tissue in biology and medicine, growth of technology use or penetration, and growth of organizations or societies including the Web [4, 35, 17]. However, our use of growth curves to model how clique tree size grows with $x = C$, $x = C/V$, or $x = E(W)$ is, to our knowledge, novel.

The Gompertz growth curve can be derived by solving the differential equation $dg(x)/dx = ag(x)$, where a is a growth coefficient [4]. Here, a is not constant but exponentially decreasing, formally $da/dx = -ka$ for $k > 0$. These two equations can be solved to obtain (18); see [4]. While a detailed study is beyond the scope of this article, it appears plausible that these differential equations reflect, at a macroscopic level, clique tree clustering's formation of cycles in a moral graph β' along with the generation of fill-in edges. Once one cycle appears in β', there may be many cycles appearing, all needing fill-in edges. Thus, once cycle formation starts in β', a faster than exponential growth in root clique tree size $g_R(x)$ is realistic and indeed supported by previous experimental results [45]. This hyper-exponential growth is in this article captured by using Gompertz growth curves.

We emphasize that Gompertz curves do not always provide accurate models of clique tree growth. For example, the property $g'_R(x) > 0$ does not reflect reality for very small $x = C$. Consider the first few BN leaf nodes added by BPART. When there is no leaf node and $x = 0$, clearly $\mu_R(0) = V$ and $\mu_M(0) = 0$. When there is one leaf node with P parents and $x = 1$, $\mu_R(1) = V - P$ and $\mu_M(1) = S^{P+1}$. Since $\mu_R(0) > \mu_R(1)$, the contribution of the root cliques to the total clique tree size in fact decreases from $x = 0$ to $x = 1$, and clearly this is not consistent with $g'_R(x) > 0$ as follows for example from (19). The situation is similar for other small values of x, see Figure 4 for $x = 2$. However, this early stage of growth is not very interesting since the total clique tree size is extremely small and typically not a concern in applications. Consequently, we consider this issue not to be an important limitation of our approach, and we use $C/V \geq 1/2$ in our experiments below.

Finally, we note that the Gompertz growth curve has a linear form, defined as follow [35].

Definition 23 (Gompertz linear form) *The Gompertz linear form is*

$$\ln\left(-\ln\frac{g(x)}{g(\infty)}\right) = \ln(\zeta) - \gamma x \tag{21}$$

Using (21), the Gompertz curve parameters ζ and γ in (18) can be estimated from data using linear regression, as we will see in Section 5. Other growth curves, including logistic and Complementary Gompertz, have forms similar to (18) that are also useful for parameter estimation by means of linear regression [35].

3.3 Growth Curves for General Bayesian Networks

We now briefly discuss the generalization from bipartite BNs, considered earlier in this section, to arbitrary BNs. There are at least two ways of going beyond bipartite BNs:

- Generalization of our analytical approach, which is tailored to bipartite BNs, to arbitrary BNs. There are several ways of doing this, First, one can maintain the use of two growth curves, but re-consider how cliques are assigned to them, in order to handle arbitrary BNs. Second, one can potentially go beyond two growth curves, and base analysis on a finite set of growth curves $\{g_1(x), g_2(x), \ldots, g_k(x)\}$ where $k \geq 1$. We discuss this approach in Section 3.3.1.

- Translation of arbitrary BNs, perhaps via some intermediate form, into bipartite BNs. Such a translation can, for example, be based on the connection between Bayesian networks and factor graphs (which are bipartite graphs) [32, 19]. The resulting bipartite BNs can then be handled using the techniques discussed elsewhere in this article. We discuss this approach in Section 3.3.2.

3.3.1 Generalization using Growth Functions

We now discuss two approaches that generalize our growth curve analysis discussed earlier in Section 3. The first approach continues to use two growth curves, but generalizes their meaning. Specifically, we introduce growth curves $g_1(x)$ and $g_2(x)$. Here, $g_1(x)$ represents cliques that contain leaf nodes and their parents (similar to $g_M(x)$), while $g_2(x)$ represents the remaining cliques (similar to $g_R(x)$). Consequently, $g_2(x)$ represents the growth of not only cliques containing root nodes only, but also cliques containing trunk nodes only, as well as cliques containing both trunk and root nodes. Clearly, this is a rather straightforward generalization approach, which may be too simple in some contexts, leading us to introduce the following alternative.

The second approach amounts to introducing an arbitrary number of growth curves. We consider a clique tree $\boldsymbol{\Gamma}$ generated from an arbitrary BN by clique tree clustering. One way to formalize the partitioning of cliques in a clique tree $\boldsymbol{\Gamma} = \{\gamma_1, \ldots, \gamma_\eta\}$ is by means of coloring the nodes in a graph (for us, a BN or a clique tree) as follows.

Definition 24 (Graph coloring) *Let $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ be a graph, let $\Phi = \{1, \ldots, \phi\}$ be a set of colors, and let h : $\boldsymbol{V} \rightarrow \Phi$ be a map (or coloring) from nodes to colors. Then $(\boldsymbol{G}, \Phi, h)$ forms a graph coloring.*

The coloring defines a partitioning of a graph's nodes into ϕ partitions. Definition 24 applies to both directed graphs (including DAGs) and undirected graphs (including clique trees). For BNs we will abuse notation slightly by saying that (β, Φ, h) is a graph coloring when $\beta = (\boldsymbol{X}, \boldsymbol{E}, \boldsymbol{P})$ is a BN; strictly speaking the coloring is in this case only for the DAG part $(\boldsymbol{X}, \boldsymbol{E})$ of the BN.

The following definition of a coloring h partitions nodes into root nodes and non-root nodes.

Definition 25 *Let $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ be a DAG and let $\Phi = \{1, 2\}$ be a set of colors. The coloring h : $\boldsymbol{V} \rightarrow \Phi$ reflects the root versus non-root status for any $V \in \boldsymbol{V}$, and is defined as*

$$h(V) = \begin{cases} 1 \text{ if } i(V) = 0 \\ 2 \text{ if } i(V) > 0 \end{cases}.$$

Similar to Definition 25, one can define a coloring that partitions nodes into leaf nodes and non-leaf nodes.

How does graph coloring apply to BNs and their clique trees? A clique in a clique tree of a BN β consists of one or more BN nodes, and these nodes may or may not have different colors as induced by a graph coloring (β, Φ, h). To reflect this, we introduce the concept of a color combination for a coloring, and have the following obvious result.

Proposition 26 *Let $(\boldsymbol{G}, \Phi, h)$ be a graph coloring and let $\phi = |\Phi|$. The number of (non-empty) color combinations is*

$$\kappa(\phi) = 2^\phi - 1.$$

Similar to Definition 19, we partition the cliques, now according to color combinations. Formally, this amounts to forming subsets of cliques Γ_i for $1 \leq i \leq \kappa(\phi)$ such that $\Gamma = \Gamma_1 \cup \cdots \cup \Gamma_{\kappa(\phi)}$ and $\Gamma_i \cap \Gamma_j = \varnothing$ for $i \neq j$. In the bipartite special case discussed in Section 3.2, $\kappa(\phi) = 2$ and $\Gamma = \Gamma_1 \cup \Gamma_2$, where Γ_1 are the root cliques and Γ_2 the mixed cliques. Assuming that BNs are randomly distributed, we let K_i be the random size of the cliques having the i-th color combination. By summing, we obtain a random variable K_T representing the total clique tree size:

$$K_T = \sum_{i=1}^{\kappa(\phi)} K_i. \tag{22}$$

This sum is a generalization of (15), which applies to the bipartite case.

For each possible color combination, and reflecting the growth of the individual random variables K_i in (22), we introduce a separate growth curve g_i with parameters $\boldsymbol{\theta}_i$ as follows.

Definition 27 *Let (G, Φ, h) be a graph coloring with $\phi = |\Phi|$ and let $g_i : \mathbb{R} \rightarrow \mathbb{R}$ be a map with parameters $\boldsymbol{\theta}_i$. Then*

$$g(x; \boldsymbol{\theta}) = \sum_{i=1}^{\kappa(\phi)} g_i(x; \boldsymbol{\theta}_i), \tag{23}$$

where $g : \mathbb{R} \rightarrow \mathbb{R}$ and $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_{\kappa(\phi)})$, is a total growth curve.

In words, Definition 27 adds up the growth curves for each color combination. A color combination corresponds to a type of clique. In this manner, we decompose the problem of estimating growth curves for complete clique trees into sub-problems of estimating growth curves for smaller pieces of clique trees. In the bipartite case, we have one color combination for mixed cliques and another color combination for root cliques, see Definition 20, resulting in two growth curves $g_R(x)$ and $g_M(x)$ on the right hand side of (23). We place no restrictions on the partitioning in Definition 24 and Definition 27, but for our purposes it typically makes sense to (i) let the coloring reflect the structure of a graph and (ii) only introduce as many colors as is needed.

3.3.2 Generalization using Factor Graphs

While our main emphasis in this article is on Bayesian networks and clique trees, many alternative approaches to representing multi-variate probability distributions by means of probabilistic graphical models exist. These alternatives include factor graphs [32, 70], Tanner graphs, Markov random fields [68, 69], and arithmetic circuits [12].

For the purpose of generalizing our growth curve approach to arbitrary BNs, factor graphs [32, 70] turn out to be of particular interest. Informally, a factor graph (FG) is a bipartite graph in which root nodes are variables, leaf nodes are factors (or functions), and a directed edge expresses an "is an argument of" relationship between a variable and a function. More formally, we have the following definition.

Definition 28 *Suppose that the function $h(X_1, \ldots, X_n)$ admits the factorization*

$$h(X_1, \ldots, X_n) = \prod_{i=1}^{m} f_i(\boldsymbol{S}_i), \tag{24}$$

where $\boldsymbol{S}_i \subseteq \{X_1, \ldots, X_n\}$ for $1 \leq i \leq m$. Then the factor graph of h is defined as a (directed) bipartite graph $(\boldsymbol{X}, \boldsymbol{F}, \boldsymbol{E})$ in which variables $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ are root nodes; $\boldsymbol{F} = \{f_1, ..., f_m\}$ are leaf nodes; and $\boldsymbol{E}$ are edges between $\boldsymbol{X}$ and $\boldsymbol{F}$ such that $(X_j, f_i) \in \boldsymbol{E}$ if and only if $X_j \in \boldsymbol{S}_i$.

Intuitively, the global function h in (24) is decomposed into products of local functions $\boldsymbol{F}$, and each local function $f \in \boldsymbol{F}$ only depends on a (hopefully small) subset $\boldsymbol{S} \subseteq \{X_1, \ldots, X_n\}$. We are interested in probabilistic inference, where $h(X_1, \ldots, X_n)$ represents a joint probability distribution over discrete random variables. Summary propagation algorithms, which are iterative algorithms that utilize "summaries" or "messages" [36], have been introduced that exploit the factor graph representation of h. Summary propagation algorithms come in two flavors, sum-product algorithms and max-product algorithms. Using sum-product algorithms, one can use factor graphs to compute marginal distributions over X_i for $1 \leq i \leq m$. The Viterbi algorithm [67, 57], generalized to arbitrary tree-structured graphical models, is a max-product algorithm called max-product belief propagation [32].

Having briefly introduced factor graphs, we now discuss how they relate to our use of bipartite BNs elsewhere in this article. Factor graphs have been extended to unify and generalize directed graphical models (Bayesian networks) and undirected graphical models (Markov networks) [19]. The studies of Bayesian networks, Markov networks, and factor graphs are therefore closely related. We now specifically exploit the close connection between factor graphs and BNs, and consider a two-step translation process (see [70] for details). First, we translate an arbitrary BN β_1 into a factor graph ϕ. Second, we translate ϕ into a bipartite BN β_2, which is related to but different from β_1. A factor graph factor from ϕ becomes a leaf node in the bipartite BN β_2; a factor graph variable from ϕ becomes a root node in β_2.

We now make a few observations related to this translation process: There are no topological restrictions on β_1; β_2 is guaranteed to be bipartite; and the size of β_2 is modest relative to β_1. This translation approach means that we can translate an arbitrary BN β_1 into a bipartite BN β_2, and then apply to β_2 the analytical and experimental machinery discussed elsewhere in this article.

4 Random Graphs and Random Bayesian Networks

Similar to BNs, random graphs are founded on graph theory. Random graphs were explored in the 1950s and early 1960s by Solomonoff and Rapoport [65] as well as Erdős and Rényi [18]. Compared to previous graph theory research, the contribution of research on random graphs was and is its emphasis on graphs as probabilistic objects, which is similar to our perspective on random BNs including BPART BNs. Random graphs continue to be studied as part of graph theory in the pure mathematics [7], but interesting connections have also been made to applied areas including social networks; spread of disease; spread of information; and information search in the World Wide Web [49]. While a comprehensive discussion of random graphs is beyond the scope of this article, we now briefly discuss the connection between random graphs and random BNs, and in particular random BNs as generated by the BPART algorithm.

Two prominent random graph models over n vertices are denoted $G(n, p)$ and $G(n, m)$ respectively. Both models are concerned with undirected graphs. In $G(n, p)$, each of the $\binom{n}{2}$ edges is included with a probability p. In $G(n, m)$, exactly m edges are added uniformly at random, from among all $\binom{n}{2}$ edges, without replacement. The sets of graphs defined by $G(n, p)$ and $G(n, m)$ form probability spaces.

It turns out that many random graph models, including $G(n, p)$ and $G(n, m)$, are in several respects similar. The number of edges in $G(n, p)$ clearly follows a Binomial distribution, and so the expectation is $p\binom{n}{2}$. If $m \approx p\binom{n}{2}$ then $G(n, p)$ and $G(n, m)$ behave, in many respects, the same [7]. In particular, there is often an emergence of

certain global properties — including graph structures in the form of trees, cycles, and cliques — as local connectivity parameters such as p (for the $G(n,p)$ model) or m (for the $G(n,m)$ model) increase for a fixed or varying n. How is a property likely to emerge? Erdős and Rényi studied $p(n)$ as $n \to \infty$ and found that emergence is often fast. In other words, many properties quickly go from very unlikely to very likely; there is a phase transition at a certain probability $p(n)$.

Random graphs can be studied from an evolutionary perspective as well. In this perspective, edges are added one by one, starting with zero edges in the random graph and approaching a fully connected graph with $\binom{n}{2}$ edges as $p \to 1$. (Clearly, as will be further discussed below, there is a similarity between increasing p in the $G(n,p)$ model and increasing the C/V-ratio in the BPART model.) Early on in random graph evolution, edges are isolated, and isolated components form. There are no cycles, because edges are initially likely to merge components rather than create cycles. Gradually, some unicycles and trees show up, however the largest components are only $\log n$ in size. Then, all of a sudden, comes the so-called double-jump [18] where two things happen. First, the number of cycles increases dramatically. Second, the size of the largest component grows, and the growth depends on whether $c < 1$ or $c > 1$. With $p = \frac{c}{n}$ and $c < 1$, the random graph consists of small components, the largest of size $\Theta(\log n)$. For $c > 1$, on the other hand, many of these small components have clustered into a "giant" component of size $\Theta(n)$. In other words, the giant component emerges when the average node degree is $np = 1$.

Which, if any, of the random graph models $G(n,p)$ and $G(n,m)$ is most relevant to the work discussed in this article? BPART(V, C, P) is, under certain conditions which we discuss shortly, very similar to $G(n,m)$. The conditions are as follows:

1. For BPART(V, C, P), the moral edges between the root nodes (induced by leaf nodes), or in other words the undirected moral graph induced by the root nodes, is what is important for our purposes, and we put $n = V$.

2. In BPART(V, C, P), we consider $P = 2$, since this yields independence between moral edges, similar to how edges are picked independently under both $G(n,p)$ and $G(n,m)$.

To make the connection between the $G(n,m)$ model and BPART more explicit, we set $n = V$ and $P = 2$, as stated in the two conditions above, and we may say BPART(n, C) instead of BPART(V, C, P).

Now, the only difference between $G(n,m)$ and BPART(n, C) is that the former picks edges without replacement, while the latter picks edges with replacement. However, this difference may not be that important in some situations. If $\binom{n}{2} >> m$, then this difference can often, for approximation purposes, simply be ignored and we set $m = C$. Alternatively, if the difference cannot be ignored, one may proceed as follows. Without loss of generality, we assume that m and n are fixed in $G(n,m)$, and consider $E(W; C, V)$ as given by (12) or (13) after substituting $V = n$. In the case of (12), we can then obtain the integer-valued lower bound C_ℓ and upper bound C_u, where $E(W; C_\ell, V) \leq m \leq E(W; C_u, V)$ and C_ℓ = C_u - 1. We then use C_ℓ and C_u as input parameter to BPART, and use BPART(n, C_ℓ) and BPART(n, C_u) to bound $G(n,m)$. If instead of (12) we use (13), one can put $E(W) = m$ and then solve for C in (13). Obviously, the solution for C will not in general be integer-valued, so one can use $C_\ell = \lfloor C \rfloor$ and $C_\ell = \lceil C \rceil$ as inputs to BPART in a manner similar to above.

The issue of the treewidth of random graphs has, to our knowledge, not been extensively researched. However, there are some results, which we now briefly review [30]. The following two analytical results apply to the early evolution of random graphs, before the giant component emerges.

Lemma 29 *Let $0 < c < 1$ and suppose that $p = c/n$. Then almost every graph $G(n,p)$ is such that every connected component is a tree or a unicycle graph.*

Corollary 30 *If $m < \frac{n}{2}$, then almost every graph $G(n,m)$ has treewidth at most two.*

With respect to the above Lemma's application to BPART, we note that triangulation of trees and unicycle graphs is quite simple. Trees do not need any fill-in edges, of course, while triangulation of a unicycle graph with k nodes amounts to adding $\lceil \frac{k}{2} \rceil$ fill-in edges. A consequence of Corollary 30 for BPART is that $C < V/2$, or $C/V < 1/2$, is not very interesting from a treewidth perspective.

We now state a result that applies for a broader range of random graphs [30], including after the emergence of the giant component.

Theorem 31 *Let $\delta \geq 1.18$. Then almost every graph $G(n,m)$ with $m \geq \delta n$ has treewidth $\Theta(n)$.*

Comparing $G(n,m)$ and BPART(V, C), m corresponds to C (the number of leaf nodes) while n corresponds to V (the number of root nodes). The condition $m \geq \delta n$ in Theorem 31 thus corresponds to $C \geq \delta V$, or $C/V \geq \delta$. It is

clear from Corollary 30 and Theorem 31 that $C/V \approx 1$ is an interesting region in the setting of clique tree clustering for BNs, assuming the BPART(V, C) model.

To summarize, it is clear that several fruitful connections can be made between random graphs and random BNs, and a few have been made above. In particular, we hypothesize that there is a connection between the onset of the rapid growth phase observed for BPART BNs and $\delta \approx 1.18$ in Theorem 31, and that this behavior can be modelled using large values for $g'(x)$ of a restricted growth curve $g(x)$ such as a Gompertz growth curve. At the same time, there are many caveats concerning the use of analytical results for random graphs in the analysis of random BNs; here are some of them. We start by identifying two structure-related issues. First, even if we ignore the small difference between $G(n, m)$ and BPART(V, C, 2) identified above for a moment, it is clear that $G(n, m)$ makes independence assumptions that are not made by BPART(V, C, P) for $P > 2$. Second, while there has been some work on treewidth of random graphs, as discussed briefly above, optimal maximal clique or treewidth has not been a central topics in random graphs. And even if it were, and if we take a strictly graph-theoretic perspective, the issue of optimizing the maximal clique of random graphs is not the same as optimizing total clique tree size, even though they clearly are related. Total clique tree size has previously been emphasized for BNs [28, 29]. Third, random graphs are typically considered in the limit $n \rightarrow \infty$, while this is generally not the situation for random BNs and BNs in general. Fourth, random graphs are strictly graph-theoretic, and one does not consider state spaces, which can exhibit important variation in BNs. To make further progress on understanding clique tree growth for the BPART model despite these limitations in applying results from the theory of random graphs, we now turn to our experimental results.

5 Experiments

In the experiments we address the following questions in the context of bipartite BNs sampled using BPART: How well do Gompertz growth curves match sample data in the form of clique trees generated using tree clustering, when the independent parameter as well as the nature of the sample data points are varied? How well do Gompertz growth curves fit sample data compared to alternative growth curve models? In answering these questions, we extend and complement previous experimental results [45] by: (i) introducing restricted growth curves, including Gompertz growth curves, in addition to sample means and unrestricted exponential growth curves; (ii) using a greater range of values for C/V; (iii) considering both $V = 20$ and $V = 30$; (iv) investigating $x = E(W)$ in addition to $x = C/V$ as the independent parameter; and (v) using as the dependent parameter the total clique tree size $\bar{k}^*_S$ rather than the size of the optimal maximal clique $\bar{\ell}^*_S$. Clique trees were generated, for sample BNs generated using BPART as indicated below, using an implementation of the HUGIN clique tree clustering algorithm. Clique trees were optimized heuristically, using the minimum fill-in weight triangulation heuristic, as treewidth computation is NP-complete.

In the rest of this experimental section, we discuss in Section 5.1 clique tree growth in the case of BPART BNs with $V = 30$ root nodes, using $x = E(W)$. In Section 5.2 we investigate BPART BNs with $V = 20$ root nodes and consider both $x = E(W)$ and $x = C/V$. In Section 5.3 we investigate the growth of individual BPART BNs.

5.1 Comparison between Growth Models for Multiple BNs

The purpose of the first set of experiments was to compare the Gompertz growth model with a few alternatives: Exponential, logistic, and complementary Gompertz. Here, we report on Bayesian networks generated using the signature BPART(30, C, 2, 2) with varying values for C, specifically $15 \leq C \leq 600$ or $1/2 \leq C/V \leq 20$. For each C/V-level considered, 100 BNs $\{\beta_1, \ldots, \beta_{100}\}$ were sampled using BPART, and clique trees with respective sizes $\{\bar{k}^*_S(\beta_1), \ldots, \bar{k}^*_S(\beta_{100})\}$ computed using the HUGIN system.

We now present the results of the HUGIN experiments. In the top panel of Figure 8, sample means $\hat{\mu}_R(x)$ along with corresponding points from different analytical growth curves $g_R(x)$ as a function of $x = E(W)$ are presented. Sample means $\hat{\mu}_R(x)$ are obtained by averaging, for a particular x, over $\{\bar{k}^*_S(\beta_1), \ldots, \bar{k}^*_S(\beta_{100})\}$ and then deducting $g_M(x)$. Here, the Gompertz, logistic, complementary Gompertz, and exponential functions are considered for $g_R(x)$. The bottom panel of Figure 8 shows how the growth curves in the top panel were obtained using linear forms such as (21). The following Gompertz growth curve was obtained

$$g_R(x) = 2^{30} \times \exp(-19.14 \times \exp(-0.005874x)),$$

where $x = E(W)$. The parameters ζ and γ were for the other growth curves computed in a similar manner. Clearly, the Gompertz curve fits the data much better than the alternative growth curves analyzed, with $R^2 = 0.9995$ (for Gompertz) versus $R^2 = 0.9413$ (for logistic) and $R^2 = 0.9407$ (for Complementary Gompertz). The excellent fit can also easily be confirmed visually by considering the sample means along with the corresponding data points for the Gompertz curve in the top panel of Figure 8.

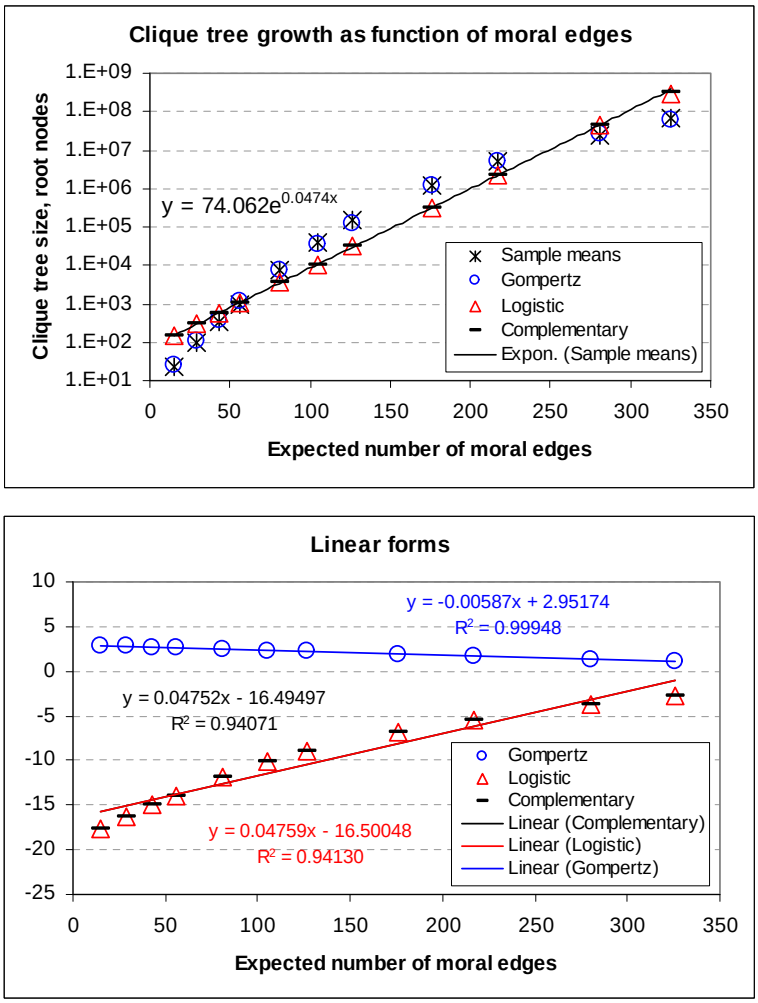

Figure 8: Experimental results for bipartite BNs with $V = 30$ root nodes and varying number of leaf nodes C. *Top:* Comparison of Gompertz and other growth curves with the sample means. The superior fit of the Gompertz curve is reflected in its better R^2 value, namely $R^2 = 0.99948$. *Bottom:* Linear forms showing how the growth curves above were obtained.

5.2 Gompertz Growth Model Details for Multiple BNs

In a second set of experiments, Bayesian networks were generated using BPART(V, C, 2, 2) with $V = 20$ and varying values for C, specifically $10 \leq C \leq 1400$ or $1/2 \leq C/V \leq 70$. Similar to above, for each C/V-level, 100 BNs were sampled using BPART, and HUGIN was used to compute clique trees with respective sizes $\{\bar{k}^*_S(\beta_1), \ldots, \bar{k}^*_S(\beta_{100})\}$. Using this relatively low value for V allowed us to generate BNs for which the generated clique trees did not exhaust the computer's memory even for very large C, thus supporting a comprehensive analysis using Gompertz growth curves with both $x = C/V$ and $x = E(W)$ as independent variables.

Figure 9 illustrates the results of these experiments. Here, the left column of Figure 9 presents Gompertz growth curves $g_R(x)$, while the right column illustrates how these growth curves were obtained using (21) similar to above. In the top row of Figure 9, sample means as well as corresponding points from a Gompertz growth curve as a function of C/V-ratio are presented. As a baseline, an exponential interpolation curve for the sample means is also provided. Empirically, the Gompertz growth curve was found to be

$$g_R(x) = 2^{20} \times \exp(-9.906 \times \exp(-0.1118x)),$$

where $x = C/V$ and with $R^2 = 0.993477$. The parameter values of $\zeta = e^{2.293} = 9.906$ and $\gamma = 0.1118$ were obtained from the Gompertz linear form as illustrated to the top right in Figure 9, based on sample means for the clique tree root cliques and the linear regression result $\ln(\zeta) - \gamma x = -0.1118x + 2.293$.

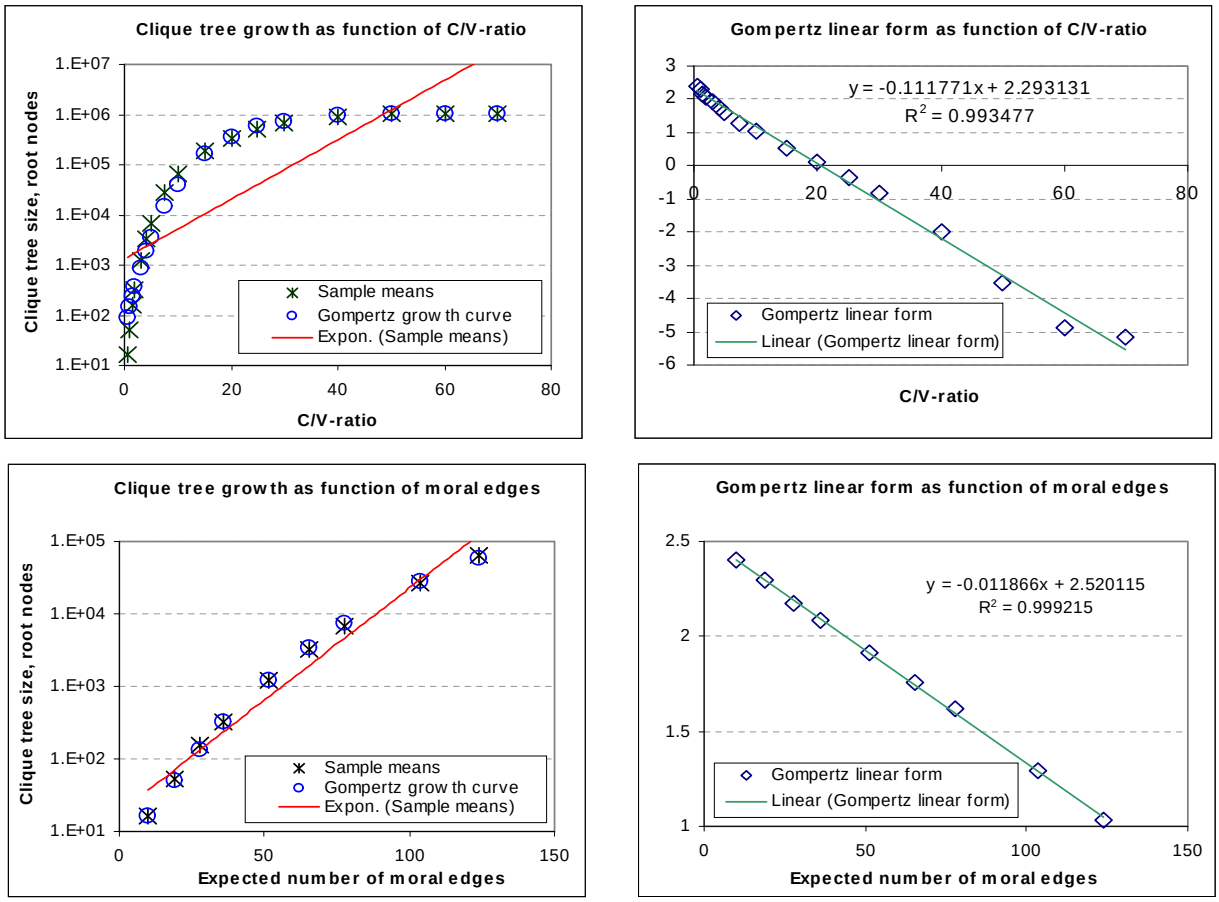

Figure 9: Empirical results for bipartite Bayesian networks generated with $V = 20$ root nodes and a varying number of leaf nodes C. *Top left:* Gompertz growth curve as a function of the C/V-ratio. *Top right:* Gompertz growth curve's linear form as a function of the C/V-ratio; used to create the Gompertz growth curve to the left. *Bottom left:* Gompertz growth curve as a function of $E(W)$. *Bottom right:* Gompertz growth curve's linear form as a function of $E(W)$; used to create the Gompertz growth curve to the left.

In the bottom row of Figure 9, we plot the expected number of moral edges $E(W)$ along the x-axis. Note that the right-most sample average in the bottom row of Figure 9, at $x = E(W) \approx 123$, corresponds to the sample average at $C/V = 10$ in the top row of Figure 9. In other words, the use of $x = C/V$ allows us to illustrate a broader range of behavior, through the inclusion of BNs with a larger number of leaf nodes, compared to when $x = E(W)$ is used. We present sample means along with the corresponding points from a Gompertz growth curve as a function of $E(W)$; an exponential regression curve is presented as a baseline. Here, the Gompertz growth curve was empirically determined to be

$$g_R(x) = 2^{20} \times \exp(-12.43 \times \exp(-0.01187x)),$$

where $x = E(W)$ and with $R^2 = 0.999215$. The parameters ζ and γ were computed in a similar manner to above and as summarized to the bottom right in Figure 9.

We now revisit the three broad growth stages discussed in Section 3 and Section 4 in terms of Figure 9. The sample means show an easy-hard-harder pattern, or monotonically increasing clique tree sizes, along these stages. The *initial growth stage*, where the C/V-ratio is "low" (for $P = 2$, up to approximately $C/V \approx 1$), is characterized by "few" leaf nodes relative to the number of root nodes. The initial growth stage is in fact difficult to see in Figure 9, since there are only a few sample points for this stage and they are very close to each other. In the *rapid growth stage*, the C/V-ratio is "medium" (for $P = 2$, from approximately $C/V \approx 1$ to say $C/V \approx 20$) and non-trivial root cliques appear. As can be seen from the sample means to the left in Figure 9, growth is initially faster than indicated by the exponential regrsssion curve and then slows down. Clearly, the Gompertz growth curves give much better fits than the respective exponential curves for both C/V and $E(W)$. The *saturated growth stage*, where the C/V-ratio is "high", is characterized by slow or no growth due to saturation. At saturation, there is one root clique γ with $|\Omega_\gamma| = 2^{20}$, hence there is no room for further growth. In Figure 9, we may say that saturation starts at $C/V \approx 20$.

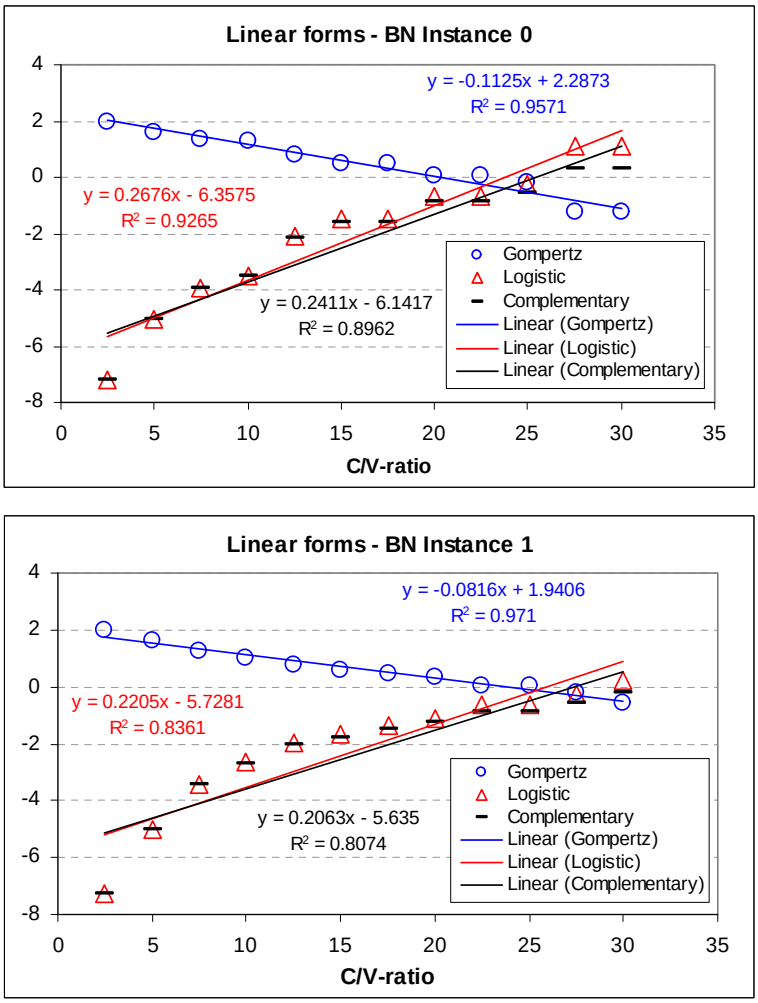

Figure 10: Experimental results for sequences of individual bipartite BNs with $V = 20$ root nodes and a varying number of leaf nodes. Comparison of Gompertz and other growth curves, as a function of C/V, is shown for two sequences of BNs. *Top:* The superior fit of the Gompertz curve for one sequence of BNs is reflected in a higher R^2 value, namely $R^2 = 0.9571$. *Bottom:* The superior fit of the Gompertz curve for another sequence of BNs is reflected in a higher R^2 value, namely $R^2 = 0.971$.

Figure 9 clearly shows the improved fit provided by Gompertz curves compared to exponential curves. Further, $x = E(W)$ provides a better fit than $x = C/V$ but for a narrower domain.

5.3 Comparison between Growth Models for Individual BNs

The experimental results so far in this section have been based on constructing growth curves $g_R(x)$ using sample means $\hat{\mu}_R(x)$ of clique tree sizes for 100 BNs per C/V-value. What happens when individual BNs, instead of multiple BNs, are used to construct growth curves $g_R(x)$? To investigate this question, we considered in a third set of experiments BNs generated using the signature BPART(20, C), with C varying from $C = 100$ to $C = 1200$. The following protocol was followed in order to create a sequence of closely related BNs. Starting with a sampled BPART(20, 1200) BN, 100 leaf nodes were deleted at a time, giving a sequence of BNs consisting of a BPART(20, 1100) BN, a BPART(20, 1000) BN, and so forth, down to a BPART(20, 100) BN. Obviously, in a real development setting the sequence of BNs might be quite different than what we used here, and in particular a machine learning algorithm or a knowledge engineer might start with a small BN and grow it, rather than the other way around. The manner in which the sequence of BNs is created for our experimental purposes does not matter as long as they are all BPART BNs, which they clearly are here.

Experimental results for two sequences of clique trees generated from the two sequences of BNs, generated accord-

ing to the above protocol, are presented in Figure 10. For the β_0 sequence (top of Figure 10), the regression results for $\bar{k}_S^*$ are: Gompertz curve $g_R(x) = -0.1125x + 2.2873$ and $R^2 = 0.9571$; Logistic curve $g_R(x) = 0.2676x - 6.3575$ and $R^2 = 0.9265$; and Complementary Gompertz curve $g_R(x) = 0.2411x - 6.1417$ and $R^2 = 0.8962$. For the $\beta_1$sequence (bottom of Figure 10), the regression results for $\bar{k}_S^*$ are: Gompertz curve $g_R(x) = -0.0816 + 1.9406$ and $R^2 = 0.971$; Logistic curve $g_R(x) = 0.2205x - 5.7281$ and $R^2 = 0.8361$; and Complementary Gompertz curve $g_R(x) = 0.2063x - 5.635$ and $R^2 = 0.8074$.

This figure clearly shows the better fit provided by Gompertz curves compared to a few alternatives. The better fit is reflected in the higher R^2 values for the Gompertz curves for both sequences. We note that the R^2 values found here, for the Gompertz curves, are smaller than the R^2 values for the Gompertz curves found in Section 5.1 and Section 5.2. A key point in this regard is that each data point here represents the clique tree size of a single BN, while each data point in Section 5.1 and Section 5.2 represents the sample mean clique tree size for 100 BNs. The poorer fit reported here is therefore not surprising.

6 Conclusion and Future Work

Substantial progress has recently been made, both in the area of Bayesian network (BN) reasoning algorithms and in the area of applications of BNs. Based on experience from applications, it is clear that Bayesian networks are useful and powerful but some care is needed when constructing them. In particular, due to the inherent computational complexity of most interesting BN queries [10, 63, 61, 1], one may want to carefully consider the issue of scalability when generating BNs for resource-bounded systems including real-time and embedded systems [48, 40]. BN generation may be performed manually, by means of knowledge-based model construction, or using machine learning methods. In resource-bounded systems, BN compilation approaches including clique tree propagation [33, 2, 25, 62] and arithmetic circuit propagation [12, 9, 8] are of particular interest [43]. In the clique tree approach, which we emphasize in this article, BN inference consists of propagation in a clique tree that is compiled from a Bayesian network. Total clique tree size is important because it determines the inference time. Unfortunately, a precise understanding of how varying structural parameters in BNs causes variation in the sizes of the induced clique tree sizes has been lagging. To attack this problem, we have in this article investigated the clique tree clustering approach, using bipartite BNs sampled by means of the BPART algorithm, by employing restricted and unrestricted growth curves. We have characterized the growth of clique tree size as a function of (i) the expected number of moral edges or (ii) the C/V-ratio, where C is the number of leaf nodes and V is the number of non-leaf nodes. In this article, we varied both (i) and (ii) by increasing the number of leaf nodes in our bipartite BNs, and also discussed how the approach applies to arbitrary BNs. Gompertz growth curves have, for the bipartite BNs investigated, been shown to give excellent fit to empirical clique tree data and they appear theoretically plausible as well.

The growth curve approach presented in this article and in an earlier paper [41] is novel and extends previous work [45]. We consider the expected number of moral edges $E(W)$ as well as the C/V-ratio, and a wide range of C/V-ratio values. We focus on the total clique tree size as opposed to size of the largest clique in the clique tree. We believe that the research reported here helps to fill a gap that appears to exist between theoretical complexity results and empirical results for specific algorithms and application BNs. To fill this gap, we have here presented an approach that combines probabilistic analysis, restricted growth curves, and experimentation. Analytically and experimentally, we have shown that the restricted growth curves induce three stages for growing Bayesian networks: The initial growth stage, the rapid growth stage, and the saturated growth stage. These stages are similar to what has been found for the evolution of random graphs. Our growth-curve results provide more detail compared to pure complexity-theoretic results; however they admittedly gloss over details available in the raw experimental data.

Areas for future work include the following. First, this type of approach may be utilized in trade-off studies for the design of vehicle health management systems including diagnostic reasoners [40], in the analysis of knowledge-based model construction algorithms, and perhaps even in the study of machine learning. In all these cases there is uncertainty regarding the impact of different BN structures on clique tree size (and consequently computation time). In knowledge-based model construction, BNs are constructed dynamically, while during the early design of health management systems there may be little information available concerning the vehicle being developed. In machine learning, especially in structure learning, one could during model selection score BNs according to their estimated computational feasibility in addition to their statistical fit. Second, these analytical growth curves can also used to perform forecasts and derive requirements for very large-scale BNs, which may have clique trees larger than what current software or hardware are capable of supporting. Third, it would be natural to develop more fine-grained analytical models, including more accurate models for arbitrary number of parents, perhaps by improving our analytical growth models based on more extensive experimentation. Finally, further exploration of the connection between random BNs, random graphs, and BNs from applications would also be interesting.

Acknowledgments

This material is based upon work supported by NASA under awards NCC2-1426, NNA07BB97C, and NNA08205346R as well as NSF awards CCF0937044 and ECCS0931978. Comments from the anonymous reviewers, which helped improve the article, are also acknowledged.

References

[1] A. M. Abdelbar and S. M. Hedetnieme. Approximating MAPs for belief networks is NP-hard and other theorems. *Artificial Intelligence*, 102:21–38, 1998.

[2] S. K. Andersen, K. G. Olesen, F. V. Jensen, and F. Jensen. HUGIN—a shell for building Bayesian belief universes for expert systems. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, volume 2, pages 1080–1085, Detroit, MI, August 1989.

[3] S. Arnborg, D. G. Corneil, and A. Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM Journal of Algebraic and Discrete Methods*, 8:277–284, 1987.

[4] R. B. Banks. *Growth and Diffusion Phenomena*. Springer, New York, 1994.

[5] A. Becker and D. Geiger. Approximation algorithms for the loop cutset problem. In *Proceedings of the Tenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-94)*, pages 60–68, San Francisco, CA, 1994.

[6] T. W. Bickmore. A probabilistic approach to sensor data validation. In *AIAA, SAE, ASME, and ASEE 28th Joint Propulsion Conference and Exhibit*, Nashville, TN, 1992.

[7] B. Bollobas. *Random Graphs*. Cambridge University Press, 2001.

[8] M. Chavira. *Beyond Treewidth in Probabilistic Inference*. PhD thesis, University of California, Los Angeles, 2007.

[9] M. Chavira and A. Darwiche. Compiling Bayesian networks using variable elimination. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07)*, pages 2443–2449, Hyderabad, India, 2007.

[10] F. G. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42:393–405, 1990.

[11] A. Darwiche. Recursive conditioning. *Artificial Intelligence*, 126(1-2):5–41, 2001.

[12] A. Darwiche. A differential approach to inference in Bayesian networks. *Journal of the ACM*, 50(3):280–305, 2003.

[13] A. P. Dawid. Applications of a general propagation algorithm for probabilistic expert systems. *Statistics and Computing*, 2:25–36, 1992.

[14] R. Dechter. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113(1-2):41–85, 1999.

[15] R. Dechter and Y. El Fattah. Topological parameters for time-space tradeoff. *Artificial Intelligence*, 125(1-2):93–118, 2001.

[16] R. Dechter and J. Pearl. Network-based heuristics for constraint satisfaction problems. *Artificial Intelligence*, 34(1):1–38, 1987.

[17] D. M. Easton. Gompertzian growth and decay: A powerful descriptive tool for neuroscience. *Physiology & Behavior*, 86(3):407 – 414, 2005.

[18] P. Erdos and A. Renyi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5:17–61, 1960.

[19] B. J. Frey. Extending factor graphs so as to unify directed and undirected graphical models. In *Proc. of the 19th Conference in Uncertainty in Artificial Intelligence (UAI-03)*, pages 257–264, 2003.

[20] R. G. Gallager. Low density parity check codes. *IRE Transactions on Information Theory*, 8:21–28, Jan 1962.

[21] F. Hutter, H. H. Hoos, and T. Stützle. Efficient stochastic local search for MPE solving. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05)*, pages 169–174, Edinburgh, Scotland, 2005.

[22] J. S. Ide and F. G. Cozman. Generating random Bayesian networks. In *Proceedings on 16th Brazilian Symposium on Artificial Intelligence*, pages 366–375, Porto de Galinhas, Brazil, November 2002.

[23] J. S. Ide, F. G. Cozman, and F. T. Ramos. Generating random Bayesian networks with constraints on induced width. In *Proceedings of the 16th European Conference on Artificial Intelligence*, pages 323–327, 2004.

[24] T. S. Jaakkola and M. I. Jordan. Variational probabilistic inference and the QMR-DT database. *Journal of Artificial Intelligence Research*, 10:291–322, 1999.

[25] F. V. Jensen, S. L. Lauritzen, and K. G. Olesen. Bayesian updating in causal probabilistic networks by local computations. *SIAM Journal on Computing*, 4:269–282, 1990.

[26] P. Jones, C. Hayes, D. Wilkins, R. Bargar, J. Sniezek, P. Asaro, O. J. Mengshoel, D. Kessler, M. Lucenti, I. Choi, N. Tu, and J. Schlabach. CoRAVEN: Modeling and design of a multimedia intelligent infrastructure for collaborative intelligence analysis. In *Proceedings of the International Conference on Systems, Man, and Cybernetics*, pages 914–919, San Diego, CA, October 1998.

[27] K. Kask and R. Dechter. Stochastic local search for Bayesian networks. In *Proceedings Seventh International Workshop on Artificial Intelligence and Statistics*, Fort Lauderdale, FL, Jan 1999. Morgan Kaufmann.

[28] U. Kjaerulff. Triangulation of graphs: algorithms giving small total state space. Technical Report R-90-09, Department of Mathematics and Computer Science, 1990.

[29] U. Kjaerulff. Approximation of Bayesian networks through edge removals. Technical Report IR-93-2007, Department of Mathematics and Computer Science, 1993.

[30] T. Kloks. *Treewidth: Computations and Approximations*. Springer-Verlag, 1994.

[31] A. M. C. A. Koster, H. L. Bodlaender, and S. P. M. van Hoesel. Treewidth: Computational experiments. In H. Broersma, U. Faigle, J. Hurink, and S. Pickl, editors, *Electronic Notes in Discrete Mathematics*, volume 8. Elsevier Science Publishers, 2001.

[32] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.

[33] S. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems (with discussion). *Journal of the Royal Statistical Society series B*, 50(2):157–224, 1988.

[34] Z. Li and B. D'Ambrosio. Efficient inference in Bayes nets as a combinatorial optimization problem. *International Journal of Approximate Reasoning*, 11(1):55–81, 1994.

[35] J. K. Lindsey. *Statistical Analysis of Stochastic Processes in Time*. Cambridge, Cambridge, 2004.

[36] H.-A. Loeliger. An introduction to factor graphs. *IEEE Signal Processing Magazine*, 21(1):28–41, 2004.

[37] D. J. C. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, Cambridge, UK, 2002.

[38] R. J. McEliece, D. J. C. Mackay, and J.-F. Cheng. Turbo decoding as an instance of Pearl's belief propagation algorithm. *IEEE Journal on Selected Areas in Communications*, 16(2):140–152, 1998.

[39] O. J. Mengshoel. *Efficient Bayesian Network Inference: Genetic Algorithms, Stochastic Local Search, and Abstraction*. PhD thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, April 1999.

[40] O. J. Mengshoel. Designing resource-bounded reasoners using Bayesian networks: System health monitoring and diagnosis. In *Proceedings of the 18th International Workshop on Principles of Diagnosis (DX-07)*, pages 330–337, Nashville, TN, 2007.

[41] O. J. Mengshoel. Macroscopic models of clique tree growth for Bayesian networks. In *Proceedings of the Twenty-Second National Conference on Artificial Intelligence (AAAI-07)*, pages 1256–1262, Vancouver, British Columbia, 2007.

[42] O. J. Mengshoel. Understanding the role of noise in stochastic local search: Analysis and experiments. *Artificial Intelligence*, 172(8-9):955–990, 2008.

[43] O. J. Mengshoel, A. Darwiche, K. Cascio, M. Chavira, S. Poll, and S. Uckun. Diagnosing faults in electrical power systems of spacecraft and aircraft. In *Proceedings of the Twentieth Innovative Applications of Artificial Intelligence Conference (IAAI-08)*, pages 1699–1705, Chicago, IL, 2008.

[44] O. J. Mengshoel, D. Roth, and D. C. Wilkins. Portfolios in stochastic local search: Efficiently computing most probable explanations in Bayesian networks. *Accepted, Journal of Automated Reasoning*, 2010.

[45] O. J. Mengshoel, D. C. Wilkins, and D. Roth. Controlled generation of hard and easy Bayesian networks: Impact on maximal clique size in tree clustering. *Artificial Intelligence*, 170(16-17):1137–1174, 2006.

[46] O. J. Mengshoel, D. C. Wilkins, and D. Roth. Initialization and restart in stochastic local search: Computing a most probable explanation in Bayesian networks. *Accepted, IEEE Transactions on Knowledge and Data Engineering*, 2010.

[47] D. Mitchell, B. Selman, and H. J. Levesque. Hard and easy distributions of SAT problems. In *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92)*, pages 459–465, San Jose, CA, 1992.

[48] D. Musliner, J. Hendler, A. K. Agrawala, E. Durfee, J. K. Strosnider, and C. J. Paul. The challenges of real-time AI. *IEEE Computer*, 28:58–66, January 1995.

[49] M. Newman, A. L. Barabási, and D. J. Watts, editors. *The Structure and Dynamics of Networks*. Princeton University Press, 2006.

[50] A. Y. Ng and M. I. Jordan. Approximate inference algorithms for two-layer Bayesian networks. In *Advances in Neural Information Processing Systems 12 (NIPS-99)*. MIT Press, 2000.

[51] L. Otten and R. Dechter. Bounding search space size via (hyper)tree decompositions. In *Proc. of the 24th Conference on Uncertainty in Artificial Intelligence (UAI-08)*, pages 452–459, 2008.

[52] J. D. Park and A. Darwiche. Approximating MAP using local search. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pages 403–410, Seattle, WA, 2001.

[53] J. D. Park and A. Darwiche. Complexity results and approximation strategies for MAP explanations. *Journal of Artificial Intelligence Research (JAIR)*, 21:101–133, 2004.

[54] J. D. Park and A. Darwiche. A differential semantics for jointree algorithms. *Artificial Intelligence*, 156(2):197–216, 2004.

[55] J. Pearl. A constraint - propagation approach to probabilistic reasoning. In L. N. Kanal and J. F. Lemmer, editors, *Uncertainty in Artificial Intelligence*, pages 357–369. Elsevier, Amsterdam, Netherlands, 1986.

[56] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA, 1988.

[57] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77:257–286, 1989.

[58] B. W. Ricks and O. J. Mengshoel. The diagnostic challenge competition: Probabilistic techniques for fault diagnosis in electrical power systems. In *Proc. of the 20th International Workshop on Principles of Diagnosis (DX-09)*, Stockholm, Sweden, 2009.

[59] I. Rish, M. Brodie, and S. Ma. Accuracy vs. efficiency trade-offs in probabilistic diagnosis. In *Eighteenth national conference on Artificial intelligence (AAAI-02)*, pages 560–566, Edmonton, Canada, 2002.

[60] C. Romessis and K. Mathioudakis. Bayesian network approach for gas path fault diagnosis. *Journal of engineering for gas turbines and power*, 128(1):64–72, 2006.

[61] D. Roth. On the hardness of approximate reasoning. *Artificial Intelligence*, 82:273–302, 1996.

[62] P. P. Shenoy. A valuation-based language for expert systems. *International Journal of Approximate Reasoning*, 5(3):383–411, 1989.

[63] E. Shimony. Finding MAPs for belief networks is NP-hard. *Artificial Intelligence*, 68:399–410, 1994.

[64] M.A. Shwe, B. Middleton, D.E. Heckerman, M. Henrion, E.J. Horvitz, H.P. Lehmann, and G.F. Cooper. Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base: I. The probabilistic model and inference algorithms. *Methods of Information in Medicine*, 30(4):241–255, 1991.

[65] R. Solomonoff and A. Rapoport. Connectivity of random nets. *Bulletin of Mathematical Biology*, 13(2):107–117, June 1951.

[66] H. J. Suermondt and G. F. Cooper. Probabilistic inference in multiply connected belief networks using loop cutsets. *International Journal of Approximate Reasoning*, 4:283–306, 1990.

[67] A.J. Viterbi. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, 13:260–269, 1967.

[68] M. Wainwright, T. Jaakkola, and A. Willsky. MAP estimation via agreement on (hyper)trees: Message-passing and linear programming approaches. *IEEE Transactions on Information Theory*, 51:3697–3717, 2002.

[69] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree-based reparameterization framework for analysis of sum-product and related algorithms. *IEEE Transactions on Information Theory*, 49:2003, 2003.

[70] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. In *Exploring artificial intelligence in the new millennium*, pages 239–269. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[71] N. L. Zhang and D. Poole. Exploiting causal independence in Bayesian network inference. *Journal of Artificial Intelligence Research*, 5:301–328, 1996.