# 3

# ASYMMETRIC HIDDEN MARKOV MODELS

*In this chapter, we introduce asymmetric hidden Markov models, which generalize the emission distributions of HMMs to arbitrary Bayesian-network distributions, allowing for state-specific graphical structures defined over the feature space. As a consequence, HMM-As are able to render more compact state spaces, thus from a learning perspective HMM-As can better cope with model overfitting compared to other HMM architectures. We study representation properties of asymmetric and symmetric HMMs, as well as provide a learning algorithm for HMM-As. Empirical results based on simulated and real-world data from several domains show the effectiveness of modeling more general asymmetries as done by HMM-As and the insight that such models can yield.*

## 3.1 INTRODUCTION

In many dynamic systems, complex patterns of observations are emitted over time. It is often the case that parts of the underlying process are not observed, e.g. because it is too difficult or impossible. This situation imposes challenges for capturing the interactions between observable features. Hidden Markov models are often employed as models for dynamic systems, having been successful in speech recognition and synthesis domain [119, 141, 168]. HMMs have also been applied to problems such as gene prediction and biological sequences [60, 165], information retrieval [64, 155], and business processes [148]. However, it has been also recognized that HMMs might face limitations to properly capture distributions when limited data is available [13, 75, 119]. Furthermore, HMMs in practice often have a single chain of states and impose a naive structure over the feature space, which on the one hand alleviates learning and inference costs, but on the other hand gives rise to larger state spaces that can lead to learning issues (e.g. model overfitting) and unsatisfactory problem insight.

Research has been dedicated to extending HMMs for representing more structural information, aiming to render more useful and accurate models, e.g. factorial HMMs [76], hierarchical HMMs [62], HMM/BN [119], and autoregressive HMMs [139]. Nevertheless, these extensions do not capture more specialized independences, often referred to as asymmetric independences or local structure [73], i.e. independences that hold for subsets of values of the involved variables. In the context of graphical models, the representation of asymmetries dates back to Bayesian multinets [73] and similarity networks [84], and had its importance recognized for allowing better probabilistic inference [19, 32, 173], learning [66, 137], and for improving problem insight [102] as well.

In the context of HMMs, however, research on capturing asymmetries has been much more limited, with a focus mostly on autoregressive models. Such models include the representation of higher-order autoregressive interactions by means of dynamic multinets [12], tree-based interactions on the observables space by means of Chow-Liu HMMs [102], and a combination of first-order autoregressions and tree-based interactions as implemented by conditional Chow-Liu HMMs [12]. Therefore, a model able to capture more general asymmetries on the observables space is needed. The literature also lacks a better understanding of the implications of employing asymmetry models in time series settings. To address these research aspects, we propose *asymmetric hidden Markov models* (HMM-As). In HMM-As observations are emitted according to state-specific Bayesian-network distributions, thus these models are able to represent independences that are not represented in symmetric HMMs.

The contributions of this chapter are as follows. We first define HMM-As, and compare its representation aspects with families of symmetric HMMs with respect to their state space dimensions. Then, we discuss a learning algorithm for HMM-As, which is based on the structural expectation-maximization framework [53, 65], and additionally analyze computational costs associated to symmetric and asymmetric HMMs. A set of varied simulations is then presented, with special attention to the effect of different dataset sizes and number of underlying structured states when learning symmetric and asymmetric models. Finally, we discuss experiments based on real-world datasets, where we take a close look at the obtained models in order to gain additional insight supported by HMM-As. Such empirical results indicate that HMM-As can be successfully used to obtaining new insight from real-life problems from several domains, including business processes, monitoring of urban pollution, and industrial processes.

The remainder of this chapter is organized as follows. In Section 3.2 we provide basic notions on distribution asymmetries and HMMs that represent asymmetries. In Section 3.3 we define HMM-As and relate them to other HMMs. In Section 3.4 a learning procedure for HMM-As is introduced. Section 3.5 reports results based on simulated data, while Section 3.6 reports results based on real-world data and discusses problem insight. In Section 3.7 the related work is discussed. The summary and future work are discussed in Section 3.8.

## 3.2 BASIC NOTIONS

In Chapter 2, we discussed different classes of HMMs, which included the most common one, i.e., the independent HMM, as well as other HMMs. It is worth noting that the HMMs shown in Figure 2.4 do not capture *asymmetries* in the distribution, i.e. independences that are valid for some values within the domains of the variables. Such independences can be formally defined by the notion of *context-specific independences*, which we define next, based on [19].

**Definition 3.1** (Contextual independence). *Let $P$ be a probability distribution over the sets of random variables $\mathbf{V}$, $\mathbf{W}$, $\mathbf{U}$, and $\mathbf{C}$, which are pairwise disjoint. We say that $\mathbf{V}$ is* *contextually independent* *of $\mathbf{W}$ given $\mathbf{U}$ and the* *context* *$\mathbf{c} \in$* *dom*$(\mathbf{C})$ *if $(\mathbf{V} \perp\!\!\!\perp_P \mathbf{W} \mid \mathbf{c}, \mathbf{U})$ for all values of $\mathbf{V}$, $\mathbf{W}$, and $\mathbf{U}$.*

Context-specific independences are able to capture independence statements that are not captured with conditional independence statements. In the context of HMMs, the context is typically given by values of the state variables(s), and we shall refer to such statements as *asymmetric independences* in this chapter.

A summary of HMMs that represent distribution asymmetries is given in Table 3.1. For such HMMs, each state $s \in \text{dom}(S)$ determines the parent set of each observable, thus leading to asymmetric independences of the form:

$$(\mathbf{V}^{(t)} \perp\!\!\!\perp_P \mathbf{W}^{(t)} \mid s^{(t)}, \mathbf{U}) \tag{3.1}$$

where $\mathbf{V}, \mathbf{W} \subseteq \mathbf{X}$. The set $\mathbf{U}$ depends on the state $s$ at $t$, and it determines the model architecture. For example, in Chow-Liu HMMs $\mathbf{U} \subseteq \mathbf{X}^{(t)}$, whereas in dynamic multinets $\mathbf{U} \subseteq \mathbf{X}^{(0:t-1)}$.

| **Model** | **Chain of states** | **Distribution asymmetries** | **Learning** |
|---|---|---|---|
| **Dynamic Multinet** [12] | Single | Higher-order autoregressions between observables. No intra-temporal correlations. | Discriminative (classification) |
| **Chow-Liu HMM** [102] | Single | Intra-temporal interactions modeled by tree distributions. | Generative |
| **Conditional Chow-Liu HMM** [102] | Single | First-order autoregressions and tree-based intra-temporal interactions. | Generative |
| **Activator DBN** [123] | No chain | Autoregressions and intra-temporal interactions between observables. | Not available |
| **Asymmetric HMM** (this chapter) | Single | Intra-temporal interactions modeled by arbitrary Bayesian network distributions. | Generative |

Table 3.1: HMM families which represent independence asymmetries.

Representing distribution asymmetries is important for inference and learning, however, as Figure 2.4 and Table 3.1 show, research to represent asymmetries has been much narrower in the context of HMMs. This is justified by the sequential nature and the role played by hidden states in HMMs, which imposes other challenges when compared to the static case. We further discuss work on representing distribution asymmetries in HMMs as follows.

As Table 3.1 indicates, systematic approaches for learning Chow-Liu HMMs are available by means of a generative-based learning. However, the representation of state-specific asymmetries in such HMMs is limited to trees, which can be harmful especially when the feature space has more features, and thus many more structures become available. On the other hand, dynamic multinets directly model potentially longer-history correlations by means of autoregressions. Yet, no instantaneous (i.e. intra-temporal) interactions are captured, which makes them closer to the original ideas of autoregressive HMMs [99, 139] by not fully exploring the graphical structure.

The learning approach of the previous asymmetry-aware HMMs is targeted at specific tasks, namely, classification. Thus, there is a need for models that can represent more general asymmetries within the feature space, yet in a compact manner to avoid the need for large amounts of data. Furthermore, the literature lacks a better understanding of the representation capacities of the independent HMM and other, structured HMMs with respect to state space dimensions and model fit when the data generation process has varying amount of structure.

## 3.3 ASYMMETRIC HIDDEN MARKOV MODELS

Asymmetric hidden Markov models generalize HMMs by allowing the emission distributions to represent additional qualitative independence per state. In the following we define HMM-As by first defining the association between states and Bayesian-network distributions, followed by a discussion on model parameterization.

### 3.3.1 *Model specification*

In order to define asymmetries in HMMs, we consider that hidden states induce local models over the observables. This notion can be conveniently represented by conditional Bayesian networks [104], in which a distribution $P(\mathbf{X} \mid S)$ is defined for the observables $\mathbf{X}$ and the state $S$. As standard conditional BNs provide a single factorization of $\mathbf{X}$ for all $s \in \mathrm{dom}(S)$, we extend this notion for accommodating more general state-specific models as follows.

**Definition 3.2** (State-specific Bayesian network). *Let $\mathbf{X}$ and $S$ be random variables. For each $s \in dom(S)$, we associate a Bayesian network over $\mathbf{X}$ called state-specific Bayesian network for s. If $B_s = (P_s, G_s)$ is the state-specific BN associated to s, we define the following conditional distribution:*

$$P(\mathbf{X} \mid s) = P_s(\mathbf{X}) \tag{3.2}$$

$$= \prod_{i=1}^{n} P_s(X_i \mid \pi_s(X_i)) \tag{3.3}$$

*where $\pi_s(X_i)$ denotes the parent set of $X_i$ as dictated by the state-specific BN $B_s = (G_s, P_s)$, in which $G_s$ denotes its graphical structure and $P_s$ its conditional probability tables.*

The previous definitions map hidden states to BNs, thus conveniently allowing multiple sets of parents for the features in $\mathbf{X}$, one for each state-specific BN.

**Definition 3.3** (Asymmetric hidden Markov model)**.** *An asymmetric hidden Markov model over the random variables* $(\mathbf{X}, S)$ *is a dynamic system* $\lambda = (M_{\rightarrow}, M_{\downarrow}, M_0)$*, where* $M_0$ *is an initial distribution* $P(S^{(0)})$*,* $M_{\rightarrow}$ *is a transition distribution* $P(S^{(t+1)} \mid S^{(t)})$*, and* $M_{\downarrow}$ *is an emission distribution given by*

$$P(\mathbf{X}^{(t)} \mid S^{(t)}) = P_{S^{(t)}}(\mathbf{X}^{(t)}) \tag{3.4}$$

From the definitions shown above, HMM-As are able to capture more qualitative independences in their topology than HMMs. Yet, HMM-As will share a few assumptions with HMMs, namely: the Markovian property and time-invariance. A third assumption that will also hold in HMM-As establishes that the intertemporal interaction between features must occur via state variables. Hence, given these assumptions, an unrolled HMM-A over the time horizon $\{0, \ldots, T\}$ has the following joint distribution:

$$\begin{aligned} P(S^{(0:T)}, \mathbf{X}^{(0:T)}) = P(S^{(0)}) &\prod_{t=0}^{T-1} P(S^{(t+1)} \mid S^{(t)}) \\ &\cdot \prod_{t=0}^{T} \prod_{i=1}^{n} P_{S^{(t)}}(X_i^{(t)} \mid \pi_{S^{(t)}}(X_i)) \end{aligned} \tag{3.5}$$

We note that standard HMMs (see Section 2.5.2) are therefore special cases of HMM-As, since in the standard HMMs every state is associated to the same Bayesian-network structure, i.e. $G_{s_i} = G_{s_j}$ for every $s_i, s_j \in \mathrm{dom}(S)$. An HMM-A can be also visualized as a probabilistic automaton, providing an intuitive representation for states and transitions, as Example 3.1 shows.

**Example 3.1.** *On a regular basis, measurements of print quality (PQ), room temperature (RT), ink type (IT), and media type (MT) are taken for an industrial printer. An HMM-A* $\mathcal{M}_1$ *for this problem has hidden states that dictate the underlying dynamics, named 'normal', 'failing mode one', and 'failing mode two', denoted by* $s_1$*,* $s_2$*, and* $s_3$ *respectively.* $\mathcal{M}_1$ *is shown in Figure 3.1 as a probabilistic automaton, which runs by alternating taking probabilistic transitions and emitting multivariate observations* $(PQ^{(t)}, RT^{(t)}, IT^{(t)}, MT^{(t)})$ *according to the states which it traverses.*

### 3.3.2 *Parameterization*

The conditional probability table of each observable $X_i$ in HMMs has the form $P(X_i \mid S, \pi^{-}(X_i))$, where $\pi^{-}$ refers to the other parents excluding the state. On the other hand, in HMM-As observables have their parameters associated to state-specific BNs, whose CPTs do not explicitly show the states. Nevertheless, CPTs in the standard sense can easily be obtained from HMM-As, as illustrated next.

**Example 3.2.** *In the HMM-A* $\mathcal{M}_1$ *(see Example 3.1), the conditional probability tables that are rendered for its feature set are shown in Figure 3.2.*

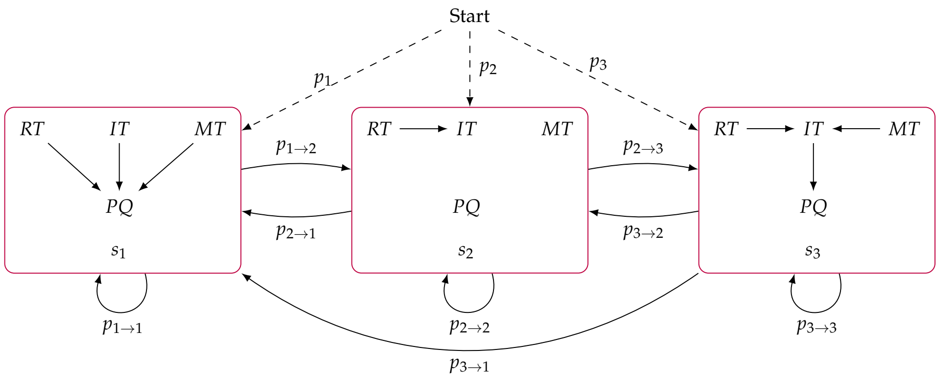


Figure 3.1: Probabilistic automaton representation for HMM-A $\mathcal{M}_1$ (dashed arcs indicate initial transitions; zero probabilities are not shown). State-specific BNs are shown in rounded rectangles.

| $\pi$ | **Parameters** |
|---|---|
| $s_1, RT, IT, MT$ | $\theta_{PQ\|s_1,RT,IT,MT}$ |
| $s_2$ | $\theta_{PQ\|s_2}$ |
| $s_3, IT$ | $\theta_{PQ\|s_3,IT}$ |

(a) Node $PQ$

| $\pi$ | **Parameters** |
|---|---|
| $s_1$ | $\theta_{RT\|s_1}$ |
| $s_2$ | $\theta_{RT\|s_2}$ |
| $s_3$ | $\theta_{RT\|s_3}$ |

(b) Node $RT$

| $\pi$ | **Parameters** |
|---|---|
| $s_1$ | $\theta_{IT\|s_1}$ |
| $s_2, RT$ | $\theta_{IT\|s_2,RT}$ |
| $s_3, RT, MT$ | $\theta_{IT\|s_3,RT,MT}$ |

(c) Node $IT$

| $\pi$ | **Parameters** |
|---|---|
| $s_1$ | $\theta_{MT\|s_1}$ |
| $s_2$ | $\theta_{MT\|s_2}$ |
| $s_3$ | $\theta_{MT\|s_3}$ |

(d) Node $MT$

Figure 3.2: Parameterization of probability tables for HMM-A $\mathcal{M}_1$.

Given the HMM-A $\mathcal{M}_1$, it is possible to obtain a standard HMM that represents its distribution over the feature space by turning the asymmetric independences of $\mathcal{M}_1$ into non-asymmetric independences, by taking the minimal directed acyclic graph (DAG) that includes all the dependences of the states in $\mathcal{M}_1$. Let us refer to such a model as *simulating* HMM, which is illustrated next.

**Example 3.3.** *Let $\mathcal{M}'_1$ be a standard HMM for simulating the HMM-A $\mathcal{M}_1$, such that both models have the same number of states. $\mathcal{M}_1$ includes asymmetric independences such as $(PQ \perp\!\!\!\perp RT \mid s_2)$, which does not hold neither in $s_1$ nor in $s_3$. This leads to the conditional dependence $(PQ \not\perp\!\!\!\perp RT \mid S)$, which therefore holds in the simulating HMM $\mathcal{M}'_1$. Similarly, in $\mathcal{M}_1$ it holds that IT and MT are independent in $s_1$ and $s_2$ only, hence, it must hold $(IT \not\perp\!\!\!\perp MT \mid S)$ in the simulating HMM. As a consequence, the structure of emissions in $\mathcal{M}'_1$ is denser than the state-specific ones from $\mathcal{M}_1$, as it can be noted from Figure 3.3a showing the emission structure of $\mathcal{M}'_1$.*

*It is worth noting that, e.g., the CPT for IT is $P(IT \mid S, RT, MT)$, although direct dependence between IT and $\{RT, MT\}$ exists only when S is $s_3$ in $\mathcal{M}_1$, which means* *that redundancies will exist in this CPT, as shown in Figure 3.3b. Finally, the total number of independent emission parameters in $\mathcal{M}_1$ is 24: 11 from $s_1$, 5 from $s_2$, and 8 from $s_3$. On the other hand, in $\mathcal{M}_1'$ there are 42 independent parameters, obtained by computing the size of the CPT for each variable.*

$$\begin{tikzcd}
\cdots \arrow[rr] & & S \arrow[rr] \arrow[dl] \arrow[d] \arrow[dr] \arrow[dd, bend left] & & \cdots \\
& RT \arrow[r] \arrow[dr] & IT \arrow[d] & MT \arrow[l] \arrow[dl] & \\
& & PQ & &
\end{tikzcd}$$

(a) Graphical structure (emissions only).

| CPT in $\mathcal{M}_1'$ | Parameters in $\mathcal{M}_1$ |
|---|---|
| $P(MT \mid S)$ | $\theta_{MT\|S}$ |
| $P(RT \mid S)$ | $\theta_{RT\|S}$ |
| $P(IT \mid MT, RT, s_1)$ | $\theta_{IT\|s_1}$ |
| $P(IT \mid MT, RT, s_2)$ | $\theta_{IT\|s_2,RT}$ |
| $P(IT \mid MT, RT, s_3)$ | $\theta_{IT\|s_3,MT,RT}$ |
| $P(PQ \mid RT, IT, MT, s_1)$ | $\theta_{PQ\|s_1,RT,IT,MT}$ |
| $P(PQ \mid RT, IT, MT, s_2)$ | $\theta_{PQ\|s_2}$ |
| $P(PQ \mid RT, IT, MT, s_3)$ | $\theta_{PQ\|s_3,IT}$ |

(b) Conditional probability tables.

Figure 3.3: Standard HMM $\mathcal{M}_1'$ that simulates the HMM-A $\mathcal{M}_1$.

Two points with further implications follow from Example 3.3. As HMM-As allow for savings in the representation size due to the direct representation of asymmetries in the distribution, one can readily take advantage of these for speeding up probabilistic inference. Secondly, in HMM-As where a few states induce small amounts of dependences (e.g. state $s_2$ in $\mathcal{M}_1$), the CPTs of the corresponding standard HMM will be large enough to cover the amount of dependences resulting from the union of all state-specific dependences of the original HMM-A. If there is a great disparity in the amount of asymmetries among the states of the HMM-A, the standard HMM will likely require more probabilistic parameters as well. As a consequence, standard HMMs are prone to reveal less insight into practical problems.

### 3.3.3 *Representation aspects*

In the following, we discuss how standard and independent HMMs can represent HMM-A distributions, and the effects of such procedure on the state space of the former model families.

#### 3.3.3.1 *Relationship with standard HMMs*

We provided in Example 3.3 an intuition on how to obtain a standard HMM able to simulate a particular HMM-A, i.e. an HMM that represents the same distribution over the space of observables. Intuitively, the simulating HMM is prone to have a denser structure compared to the individual states of the original HMM-A, which in the limit reaches a fully connected structure. This is the main idea used in Proposition 3.1 and its proof (see the Appendix) to show that a standard HMM can be obtained from a given HMM-A in the general case. This result also indicates that the simulating HMM does not need additional states for the simulation.

**Proposition 3.1.** *Let $\mathcal{M}$ be an asymmetric HMM with k states over the observables $\mathbf{X}$, where each $X_i \sim$ $\underline{\text{Multinomial}}$, $i = 1, \ldots, n$. Then, there exists a $\underline{\text{standard}}$ HMM $\mathcal{M}'$ with k states over the same observables which simulates $\mathcal{M}$, i.e. $P'(\mathbf{X}^{(0:T)}) = P(\mathbf{X}^{(0:T)})$, where P and $P'$ denote the joint distributions of $\mathcal{M}$ and $\mathcal{M}'$ over $\mathbf{X}$ respectively.*

Although the proof of Proposition 3.1 uses an argument based on full connectivity, this is not strictly necessary as the structure on the simulating HMM depends on the amount and form of asymmetries in the original HMM-A. Nevertheless, as Figure 3.3b shows, parameter redundancy at the level of states is likely to occur in the standard HMM, preventing inference from readily benefiting from distribution asymmetries, as such redundancies are encoded in the CPTs, which is not the case in HMM-As.

#### 3.3.3.2 *Relationship with independent HMMs*

While standard HMMs can simulate HMM-As using the same number of states, it is straightforward to see that independent HMMs are not able to do so in the general case. It turns out, however, that the simulation process becomes possible at the cost of expanding the state space of HMM-Is. Intuitively, the more general independence assertions in each state of a given HMM-A must be *broken* into multiple and naively-structured states. We show this result by means of Proposition 3.2.

**Proposition 3.2.** *Let $\mathcal{M}$ be an asymmetric HMM with k states over the observables $\mathbf{X}$, where each $X_i \sim$ $\underline{\text{Bernoulli}}$, $i = 1, \ldots, n$. Then, there exists an $\underline{\text{independent}}$ HMM $\mathcal{M}'$ with $k'$ states over the same observables, such that $\mathcal{M}'$ simulates $\mathcal{M}$ and $k' \leq k2^n$.*

It is straightforward to extend Proposition 3.2 for the more general case of multinomial observables. The proof of Proposition 3.2 (see Appendix 3.A) provides a method for simulating HMM-A distributions by means of HMM-Is, and it also shows an upper bound on the number of states required by the HMM-I. In practice, the amount of dependences per state and the numerical parameterization of the structured model can greatly vary, hence the number of states that a simulating HMM-I requires tends to be lower than the bound, although it can still be much higher than the original number of states of the original HMM-A. Nevertheless, as we further show in this work, a substantial increase in the state space can be expected when simulating lowly- and moderately-structured distributions.

## 3.4 LEARNING

In this section, we present a learning algorithm for HMM-As. We discuss computational costs for this and other families of HMMs as well.

### 3.4.1 *Learning setting*

In order to learn HMM-As, we assume that state variables are not observed and the graphical structure for emission distributions is unknown. In this case, i.e. learning under missing data and unknown structure, the likelihood function of the observed data is non-decomposable by the graphical structure [15], which makes analytical methods impossible. The structural expectation-maximization (see Section 2.6.4) is often employed in these settings, which serves as a basis for the learning procedure we develop for HMM-As.

The learning setting is score-based and is as follows. Consider a dataset $D$ of $m$ i.i.d. complete sequences, where the $i$th sequence has the form $\mathbf{x}[i]^{(0)}, \ldots, \mathbf{x}[i]^{(m_i)}$. Given an integer $k$, we aim to learn an HMM-A with $k$ states that best fits $D$. As in the structural EM, HMM-A learning is based on the idea of placing structure learning in each cycle of E and M steps. The learning procedure for learning HMM-As is described next, together with a discussion on its cost.

### 3.4.2 *Expectation step*

In the E step the current model $\lambda^{\text{old}}$ is used for computing two expected statistics: the expected occupancy of each state (denoted by $\gamma$), and the expected transitions between any two states (denoted by $\xi$). For the sake of exposition, we show derivations for the expected statistics considering a single sequence with length $\{0, \ldots, T\}$, which is straightforward to extend for multiple sequences as the sequences are assumed i.i.d. We repeat below the notation of expected statistics given in Section 2.6.2 for convenience:

$$\gamma_t(j) \overset{\text{def}}{=} P(S^{(t)} = s_j \mid D \colon \lambda^{\text{old}}) \tag{3.6}$$

$$\xi_t(i,j) \overset{\text{def}}{=} P(S^{(t)} = s_i, S^{(t+1)} = s_j \mid D \colon \lambda^{\text{old}}) \tag{3.7}$$

Based on the assumptions regarding the HMM-A topology, it is possible to show that the expected statistics can be given by means of the so-called *forward* and *backward* variables [141], denoted by $\alpha$ and $\beta$ respectively, similarly to regular HMMs:

$$\gamma_t(j) = \frac{\alpha_t(j) \cdot \beta_t(j)}{\sum_{i=1}^{k} \alpha_t(i) \cdot \beta_t(i)} \tag{3.8}$$

$$\xi_t(i,j) = \frac{\alpha_t(i) \cdot a_{ij} \cdot b_j(\mathbf{x}^{(t+1)}) \cdot \beta_{t+1}(j)}{\sum_{p=1}^{k} \sum_{q=1}^{k} \alpha_t(p) \cdot a_{pq} \cdot b_q(\mathbf{x}^{(t+1)}) \cdot \beta_{t+1}(q)} \tag{3.9}$$

where $a_{ij}$ denotes the transition probability from state $s_i$ to state $s_j$, and $b_j(\mathbf{x}^{(t+1)})$ denotes the emission probability of $\mathbf{x}^{(t+1)}$ according to state $s_j$. The forward and backward variables are defined as follows:

$$\alpha_t(j) \overset{\text{def}}{=} P(S^{(t)} = s_j, \mathbf{x}^{(0:t)} \colon \lambda^{\text{old}}) \tag{3.10}$$

$$= \left[ \sum_{i=1}^{k} \alpha_{t-1}(i) \cdot a_{ij} \right] b_j(\mathbf{x}^{(t)}) \tag{3.11}$$

$$\beta_t(i) \overset{\text{def}}{=} P(\mathbf{x}^{(t+1:T)} \mid S^{(t)} = s_i \colon \lambda^{\text{old}}) \tag{3.12}$$

$$= \sum_{j=1}^{k} a_{ij} \cdot b_j(\mathbf{x}^{(t+1)}) \cdot \beta_{t+1}(j) \tag{3.13}$$

where the basis of recursion is defined as $\alpha_0(i) = v_i b_i(\mathbf{x}^{(0)})$ and $\beta_T(i) = 1$ for all $i = 1, \ldots, k$, where $v_i$ denotes the initial probability of state $s_i$. The variables $\alpha$ and $\beta$ can be computed efficiently by means of dynamic programming, as illustrated by Proposition 3.3.

**Proposition 3.3.** *The computation of one E-step iteration for one sequence in asymmetric HMMs takes $\mathcal{O}(Tk^3n)$ time.*

It is straightforward to see that the cost of the E step in HMM-As is, in fact, the same as that of several other families of HMMs, including the independent and standard HMMs. We also note that the cost is strongly influenced by the number of states (which grows in a cubic fashion, whereas the other terms grow linearly).

### 3.4.3 *Maximization step*

In the M step, we obtain a new model $\lambda^{\text{new}}$ based on the expected statistics previously computed. However, as opposed to the standard EM, the M step for HMM-As can no longer be computed efficiently in its exact form, as the graphical structure on the feature space is unknown, which relates to the intractable problem of finding the optimal structure of a Bayesian network (see Section 2.3.2). In fact, this efficiency can only be attained by very few families of HMMs, where the independent HMMs is the main one; even some models that do not capture asymmetries, e.g. the standard HMMs, also lose this property since the structure is unknown (even though it is shared by all the states). To learn feature-space structures with reasonable quality, one often relies on approximate approaches.

In order to devise the update procedure for HMM-As, let us consider the expected score in SEM [14, 65]. The expected score for a candidate model $\lambda$ is the expectation of the complete data likelihood taken with respect to the hidden states conditional on the current model $\lambda^{\text{old}}$:

$$\begin{aligned} Q(\lambda, \lambda^{\text{old}}) &= \mathbb{E}_{s^{(0:T)}}\left[\log P(\mathbf{x}^{(0:T)}, s^{(0:T)} \colon \lambda) - \text{Pen}(\lambda) \mid \mathbf{x}^{(0:T)} \colon \lambda^{\text{old}}\right] \\ &= \sum_{s^{(0:T)}} P(s^{(0:T)} \mid \mathbf{x}^{(0:T)} \colon \lambda^{\text{old}}) \cdot \log P(\mathbf{x}^{(0:T)}, s^{(0:T)} \colon \lambda) \\ &\quad - \text{Pen}(\lambda) \end{aligned} \tag{3.14}$$

The expectation is taken with respect to the latent state. Note that $P(\mathbf{x}^{(0:T)}, s^{(0:T)} \mid \lambda)$ factorizes according to the structure of the HMM-A (see Equation 3.5), thus:

$$\begin{aligned} Q(\lambda, \lambda^{\text{old}}) &= \sum_{s^{(0:T)}} P(s^{(0:T)} \mid \mathbf{x}^{(0:T)} \colon \lambda^{\text{old}}) \\ &\quad \cdot \log\left[P(s^{(0)}) \prod_{t=0}^{T-1} P(s^{(t+1)} \mid s^{(t)}) \prod_{t=0}^{T} P(\mathbf{x}^{(t)} \mid s^{(t)})\right] \\ &\quad - \text{Pen}(\lambda) \end{aligned} \tag{3.15}$$

$$\begin{aligned} Q(\lambda, \lambda^{\text{old}}) &= \sum_{s^{(0:T)}} \log P(s^{(0)}) P(s^{(0:T)} \mid \mathbf{x}^{(0:T)} \colon \lambda^{\text{old}}) \\ &\quad + \sum_{s^{(0:T)}} \left(\sum_{t=0}^{T-1} \log P(s^{(t+1)} \mid s^{(t)})\right) P(s^{(0:T)} \mid \mathbf{x}^{(0:T)} \colon \lambda^{\text{old}}) \\ &\quad + \sum_{s^{(0:T)}} \left(\sum_{t=0}^{T} \log P(\mathbf{x}^{(t)} \mid s^{(t)})\right) P(s^{(0:T)} \mid \mathbf{x}^{(0:T)} \colon \lambda^{\text{old}}) \\ &\quad - \text{Pen}(\lambda) \end{aligned} \tag{3.16}$$

Equation 3.15 suggests that each term of the expected score can be optimized separately. The result is the parameter updating in the SEM process as follows.

#### 3.4.3.1 *Structure learning*

In Equation 3.15, we identify the term associated to the emissions as:

$$\begin{aligned} &\sum_{s^{(0:T)}} \left(\sum_{t=0}^{T} \log P(\mathbf{x}^{(t)} \mid s^{(t)})\right) \cdot P(s^{(0:T)} \mid \mathbf{x}^{(0:T)} \colon \lambda^{\text{old}}) - \text{Pen}(M_{\downarrow}) \\ &= \sum_{j=1}^{k} \sum_{t=0}^{T} \log P(\mathbf{x}^{(t)} \mid s_j^{(t)}) \cdot P(s_j^{(t)} \mid \mathbf{x}^{(0:T)} \colon \lambda^{\text{old}}) - \text{Pen}(M_{\downarrow}) \\ &= \sum_{j=1}^{k} \sum_{t=0}^{T} \gamma_t(j) \cdot \log P(\mathbf{x}^{(t)} \mid s_j^{(t)}) - \text{Pen}(M_{\downarrow}) \end{aligned} \tag{3.17}$$

The emissions term (Equation 3.17) can be further decomposed *per state*, because the state-specific networks are independent of each other. The advantage now is that *each state-specific network can be locally learned*. In this work, the penalty term is defined according to the BIC score (see Section 2.3.2). Thus, for a state $s_j$, its fraction from the emissions term is:

$$\begin{aligned}&\sum_{t=0}^{T} \gamma_t(j) \log P(\mathbf{x}^{(t)} \mid s_j^{(t)}) - \text{Pen}(M_{\downarrow}; s_j) \\ &= \sum_{t=0}^{T} \gamma_t(j) \log P(\mathbf{x}^{(t)} \mid s_j^{(t)}) - \frac{K_j \log(T+1)}{2} \qquad (3.18)\end{aligned}$$

where $K_j$ is the number of parameters in the model for state $s_j$. In HMM-As we wish to learn state-specific graphical structures in the M step, hence we run structure learning for each of the $k$ states separately. In practice, structure learning often relies on approximate methods for exploring the search space of structures in feasible time and yet providing reasonable solutions.

In this work, we consider the tabu search [77] (TS, for short) to explore the candidate space of structures, which is a polynomial-time procedure based on hill-climbing search. A TS iteration explores the neighborhood of the current solution (initially an empty network) by adding, deleting or reversing an arc from this solution. The current solution is added to the tabu list, which stores the 10 most recently explored networks, in this implementation. Furthermore, only neighborhood solutions that are not in the tabu list are added to the neighborhood set (initially empty). Once the neighborhood set has been updated, the best of its solutions is taken out and set in the next iteration as the current solution. The new current solution might not be better than the previous one, however, this is allowed for no more than 10 consecutive iterations.

During the tabu search for the $s_j$ state, the corresponding term in Equation 3.18 is used to compare candidate structures. Once the stopping criterion is reached in TS, the best structure that has been seen is returned. Stopping criteria include, e.g., testing whether the neighborhood set is empty, or testing if more than 10 iterations without improvement have passed. Given the described steps for TS, the cost of each structure learning run is bounded by a polynomial cost on the method's hyperparameters aforementioned and the number of observable features.

### 3.4.3.2 *Parameter update*

After obtaining a model structure for $\lambda$, it is possible to show that maximizing the expected score (Equation 3.15) leads to the following update formula for the transition probabilities:

$$\hat{a}_{ij} = \frac{\sum_{t=0}^{T-1} \xi_t(i,j)}{\sum_{t=0}^{T-1} \gamma_t(i)} \qquad (3.19)$$

The update of the emission probabilities, in turn, is more involved than in standard EM, as the feature space is multivariate and each feature can have other parents beyond the state variable. Furthermore, the parent set for a given feature can vary among states. Nevertheless, we can take advantage of the fact that the state-specific BNs allow us to factorize the joint distribution of the feature set **X**, thus we can update the probability tables for one variable at a time. For state $s_j$ and feature $X$, we update the corresponding probability tables as follows:

$$\hat{b}_j(X = x, \pi_j(X) = y) = \frac{\sum_{t=0}^{T} \gamma_t(j) \cdot \mathbb{1}(x^{(t)}, \pi_j(X)^{(t)} = y)}{\sum_{t=0}^{T} \gamma_t(j) \cdot \mathbb{1}(\pi_j(X)^{(t)} = y)} \tag{3.20}$$

where $\mathbb{1}$ is the indicator function. As in the case of arbitrary Bayesian networks, the cost of this calculation strongly depends on the connectivity of the network, being exponential in the number of features in the worst case. However, if the parent sets have moderate sizes, this can be very reasonable in practice.

As a final remark in learning HMM families, we note that a simpler version of this M step is needed for learning standard HMMs. In that case, structure learning is executed only once, as all the states will share a single structure. Analogously, updating the parameters in standard HMMs can still be costly due to the reasons previously discussed.

## 3.5 ASSESSMENT VIA SIMULATIONS

In this section, we aim to understand how unstructured, structured, and asymmetry-aware models cope with data generated from structured distributions. We also intend to analyze the effect of different amounts of data in model quality. To this end, we generated data from HMM-A distributions to simulate different scenarios. The model selection procedure used to learn models is described, as well as the data generation process, and finally the obtained results are discussed.

### 3.5.1 *Model selection*

In order to learn models that generalize best, we considered a model selection procedure to determine state spaces balancing complexity and overfitting avoidance as follows. Given a sequence dataset $D$, models are learned incrementally by increasing the number of states until overfitting occurs, which corresponds to the point where model score no longer increases. Model scoring is based on a 10-fold cross-validation: for each fold, a model is learned using training data (90% of the data) and its log-likelihood over validation data (the remaining data) is computed; after processing all the folds, the mean log-likelihood is taken, corresponding to the final score. To better assess learning, we learn 30 initial models for each $k$ states, and select the one that generalizes the best to represent models with $k$ states. Once the number of states has been determined, the final model $L$ is learned using the entire dataset and those initial parameters, and it is evaluated by means of 60 independent datasets (not used in learning nor validation; each independent dataset has 2,000 sequences with length 20 each).

Each learned model $L$ is evaluated by comparing likelihoods as follows. Let $R$ be the true model used to sample $D$, then we define the fit quality of $L$ as $\log \mathcal{L}_R - \log \mathcal{L}_L$, where $\log \mathcal{L}_R$ and $\log \mathcal{L}_L$ denote the mean log-likelihood of the models $R$ and $L$ over testing data respectively. This fit quality can be interpreted as the logarithm of the number of times the likelihood of the true model $R$ is in comparison with the likelihood of the learned model $L$ (in non-logarithmic scale). Hence, if the difference equals zero, it indicates that $L$ and $R$ fit equally well, while a difference larger than zero indicates that $L$ fits worse than $R$. Thus, learned models with log-likelihood difference closer to 0 are preferred. We finally note that this procedure allows us to compare models learned with different amounts of data, as they are evaluated over the same testing datasets.

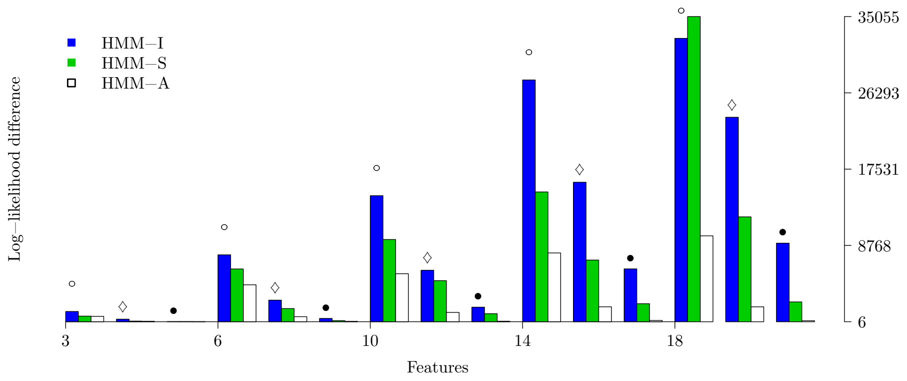
### 3.5.2 *Datasets*

Datasets were sampled from random HMM-As, which were generated taking into account that many real-life networks have an average degree between 2 and 4 per node (i.e. the sum of in- and out-degrees). This is the case, for example, in well-known BNs, such as alarm, pathfinder, asia, and insurance [151]. Hence, in order to generate ground truth models having state-specific BNs with a reasonable, and yet realistic structure, the maximum degree of each node on each network was set to 3.

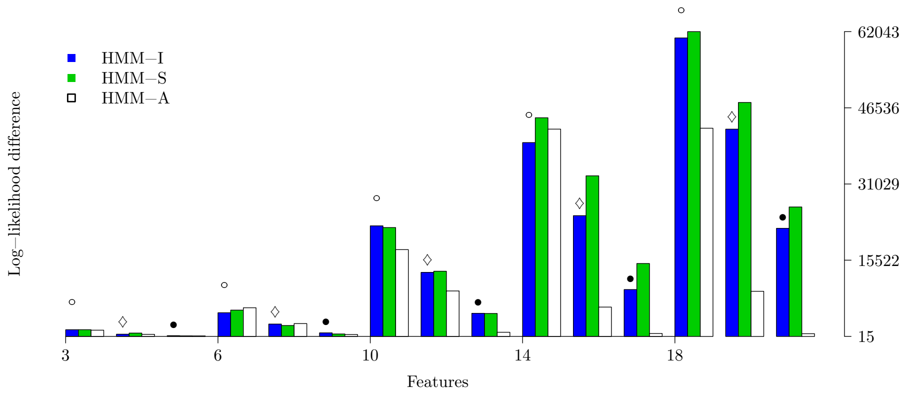
In order to build a random HMM-A with $k$ states, its initial and transition matrices are sampled from Dirichlet distributions with concentration parameters all set to 1. Thus, valid distributions are obtained, i.e. matrices with rows that sum to 1 [69]. The emissions are Bayesian networks made of uniformly sampled DAGs [122, 151], whose nodes have the aforementioned maximum degree. All observables are modeled as random variables following Bernoulli distributions, whose parameters are sampled from Dirichlet distributions as before. We note that this procedure is also used to generate the initial models used in learning (see Section 3.5.1), except that no maximum degree is set. Finally, in the constructed scenarios the following quantities were considered: number of features $n \in \{3, 6, 10, 14, 18\}$, the state space dimension of true models $k \in \{2, 6\}$, and the amount of sequential data as 50 sequences (each with length of 10 time points), 200 sequences (10 time points) and 1,000 sequences (20 time points).

### 3.5.3 *Results for symmetric models*

Figure 3.4 shows the log-likelihood differences between asymmetric and symmetric HMMs based on simulated data (here, HMM-S refers to standard HMM). We first note that, as expected, all the classes of models obtained better fit when more data is provided, which is influenced by the fact that more states can be learned prior to overfitting. The results also suggest that independent and standard HMMs were not able to provide the same model quality as HMM-As, even when the highest amounts of data were provided to all the three models. Hence, it seems reasonable to expect that much more data would be needed in order to learn models that fit as well as the learned HMM-As (in this case, using 1,000 sequences). Concerning the scarcer datasets (note that the larger datasets are 40× larger than the smaller ones), HMM-As achieved superior model quality on most cases. This allows us to conclude that HMM-As showed a good compromise in a varied range of dataset sizes, which can be explained by its flexibility on learning more or less dense feature-space structures depending on the situation.



(a) Number of states in true models = 2.



(b) Number of states in true models = 6.

Figure 3.4: Fit of asymmetric and symmetric HMMs learned in simulations. Datasets sampled from true models have 50 sequences (length 10 time points, ◦), 200 sequences (length 10 time points, ◊), and 1,000 sequences (length 20 time points, •). Note that scales on Y axes differ.

In terms of scaling, e.g. when modeling more observables, the additional structure of HMM-As avoided pitfalls that can hinder independent and standard HMMs: HMM-Is will tend to increase their state space, while standard HMMs will tend to model denser feature-space graphical structures. As a consequence, in most cases these symmetric models approach overfitting with much less model quality than HMM-As. In other words, despite the representational equivalence between HMM-As and independent and standard HMMs in theory, such symmetric models can be limited in practice. These claims are further supported by results in Table 3.2 showing the corresponding state space dimensions, and Figures 3.5a-3.5b showing the number of parameters.

Table 3.2 shows the dimension of state spaces associated to learned models, suggesting that approximating HMM-A distributions required independent HMM with state spaces substantially larger than the true models' spaces, while this was not the case for standard HMMs. Nevertheless, as Figures 3.5a-3.5b show, the number of parameters in these two families were substantially higher than those of learned HMM-As, specially when more features were involved. With regard to running time in learning, Figures 3.6a and 3.6b show that, somewhat surprisingly, learning HMM-Is was more costly in most cases than HMM-As: although learning HMM-As is done via structural EM, its combination with search heuristics and smaller space state was in practice more efficient than the EM used to learn HMM-Is.

| $n$ | HMM-I | HMM-S | HMM-A | $n$ | HMM-I | HMM-S | HMM-A |
|---|---|---|---|---|---|---|---|
| 3 | 3 | 2 | 2 | 3 | 3 | 3 | 3 |
| 6 | 3 | 2 | 2 | 6 | 3 | 2 | 2 |
| 10 | 6 | 2 | 2 | 10 | 6 | 2 | 3 |
| 14 | 5 | 2 | 2 | 14 | 7 | 2 | 4 |
| 18 | 5 | 2 | 2 | 18 | 7 | 2 | 5 |
| Number of states in true models = 2 | | | | Number of states in true models = 6 | | | |

(a) **Dataset size = 50 sequences**.

| $n$ | HMM-I | HMM-S | HMM-A | $n$ | HMM-I | HMM-S | HMM-A |
|---|---|---|---|---|---|---|---|
| 3 | 4 | 2 | 2 | 3 | 4 | 4 | 4 |
| 6 | 7 | 3 | 2 | 6 | 6 | 3 | 3 |
| 10 | 10 | 2 | 2 | 10 | 11 | 3 | 8 |
| 14 | 9 | 2 | 2 | 14 | 13 | 2 | 6 |
| 18 | 13 | 2 | 2 | 18 | 15 | 3 | 6 |
| Number of states in true models = 2 | | | | Number of states in true models = 6 | | | |

(b) **Dataset size = 200 sequences**.

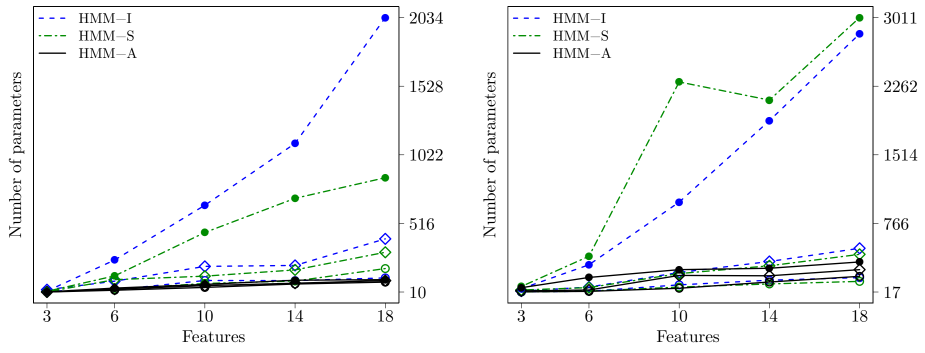
| $n$ | HMM-I | HMM-S | HMM-A | $n$ | HMM-I | HMM-S | HMM-A |
|---|---|---|---|---|---|---|---|
| 3 | 3 | 2 | 2 | 3 | 6 | 6 | 6 |
| 6 | 13 | 2 | 2 | 6 | 15 | 6 | 7 |
| 10 | 21 | 2 | 2 | 10 | 27 | 6 | 7 |
| 14 | 27 | 2 | 2 | 14 | 37 | 3 | 6 |
| 18 | 37 | 2 | 2 | 18 | 45 | 3 | 6 |
| Number of states in true models = 2 | | | | Number of states in true models = 6 | | | |

(c) **Dataset size = 1,000 sequences**.

Table 3.2: State spaces of asymmetric and symmetric HMMs learned in simulations.

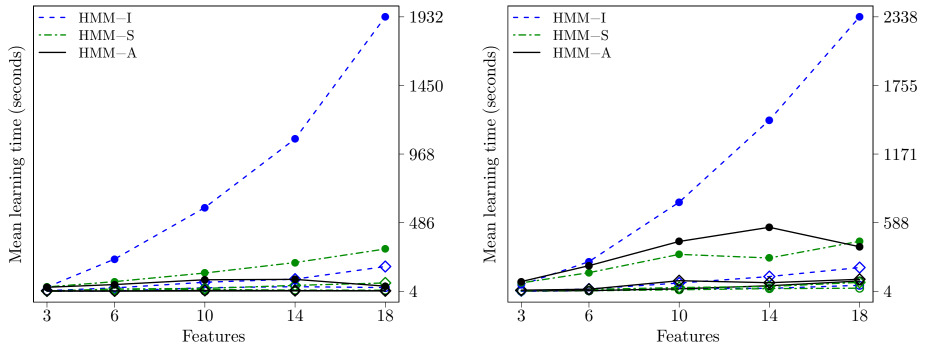
## 3.5.4 *Results for asymmetric models*

Figure 3.7 shows the fit quality results for HMM-As and Chow-Liu HMMs (HMM-CLs). These results indicate that restricting the feature space to trees



(a) Number of states in true models = 2. (b) Number of states in true models = 6.

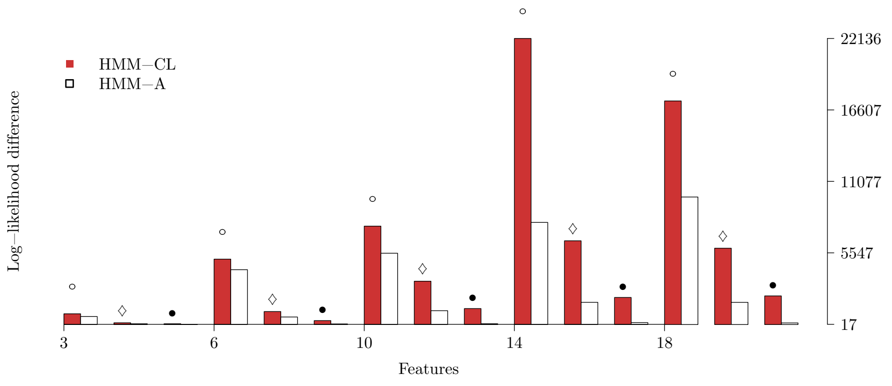
Figure 3.5: HMM-As and symmetric HMMs learned from simulated data: number of parameters for different cases.



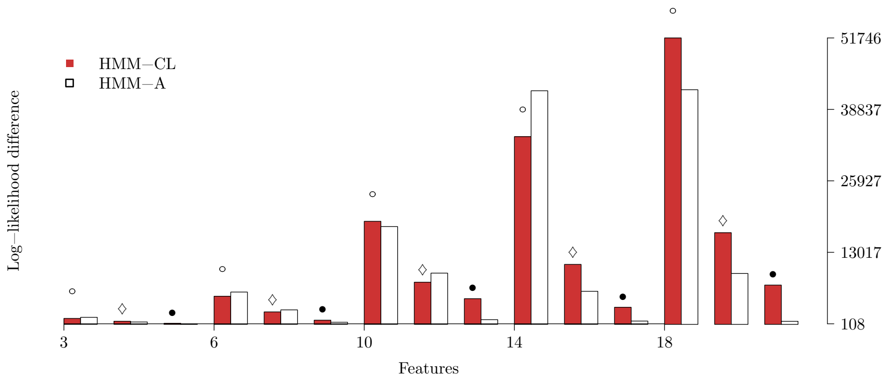
(a) Number of states in true models = 2. (b) Number of states in true models = 6.

Figure 3.6: HMM-As and symmetric HMMs learned from simulated data: mean learning time in seconds.

prevented HMM-CLs from achieving model quality as high as that by HMM-As on most of the considered scenarios. This is more evident in the cases involving more observables, where the learned HMM-As reached the most superior model quality compared to HMM-CLs, which is likely influenced by the size of the possible graphical structures for emissions, a situation which HMM-As can better handle since HMM-As are not restricted to trees. On the other hand, HMM-CLs are prone to be more efficient in practice, since learning Chow-Liu trees can be done efficiently per EM iteration [102]. Similarly to the symmetric models case, HMM-As could be trained with less data, and yet provided similar or better model quality than HMM-CLs – although here to a lesser extent when the data generating process had a higher number of hidden states. Furthermore, extending the state spaces of HMM-CLs resulted in better models, however, prior to overfitting these achieved lower quality than HMM-As.



(a) Number of states in true models = 2.



(b) Number of states in true models = 6.

Figure 3.7: Fit of asymmetric and Chow-Liu HMMs learned in simulations. Datasets sampled from true models have 50 sequences (length 10 time points, ∘), 200 sequences (length 10 time points, ◊), and 1,000 sequences (length 20 time points, •). Note that scales on Y axes differ.

A comparison based on Figures 3.4 and 3.7 suggests that modeling state-specific structures, whether by means of general asymmetries as HMM-As do or tree-shaped ones as HMM-CLs do, led to better results than those of symmetric models. HMM-As needed in general fewer states or fewer parameters than symmetric HMMs, which also holds for HMM-CLs with respect to symmetric HMMs, as shown in Table 3.3 and Figure 3.8. Hence, the results of this section suggest a somewhat consistent conclusion: capturing the distribution underlying data generated by more structured processes is more adequate by means of models that capture distribution specificities associated to the hidden states. Although symmetric models can in theory capture such distributions, whether by an increase of their state spaces or by modeling denser emissions structure, in many realistic situations – where data is often limited – the asymmetric models exhibited several advantages and better handled the complexity *versus* quality trade-off.

| $n$ | HMM-CL | HMM-A | $n$ | HMM-CL | HMM-A |
|---|---|---|---|---|---|
| 3 | 2 | 2 | 3 | 3 | 3 |
| 6 | 2 | 2 | 6 | 2 | 2 |
| 10 | 2 | 2 | 10 | 3 | 3 |
| 14 | 3 | 2 | 14 | 4 | 4 |
| 18 | 2 | 2 | 18 | 4 | 5 |
| $k$ in true models = 2 | | | $k$ in true models = 6 | | |

(a) **Dataset size = 50 sequences**.

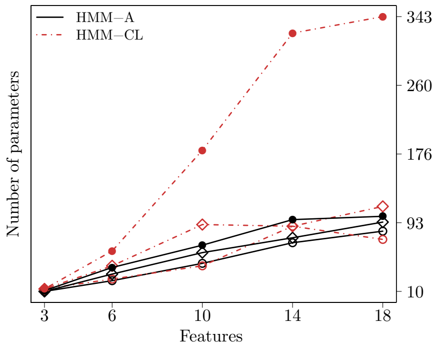
| $n$ | HMM-CL | HMM-A | $n$ | HMM-CL | HMM-A |
|---|---|---|---|---|---|
| 3 | 2 | 2 | 3 | 4 | 4 |
| 6 | 3 | 2 | 6 | 3 | 3 |
| 10 | 4 | 2 | 10 | 6 | 8 |
| 14 | 3 | 2 | 14 | 6 | 6 |
| 18 | 3 | 2 | 18 | 7 | 6 |
| $k$ in true models = 2 | | | $k$ in true models = 6 | | |

(b) **Dataset size = 200 sequences**.

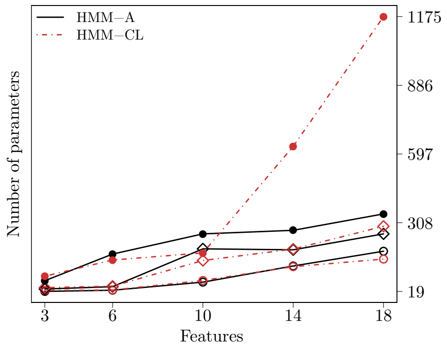
| $n$ | HMM-CL | HMM-A | $n$ | HMM-CL | HMM-A |
|---|---|---|---|---|---|
| 3 | 2 | 2 | 3 | 7 | 6 |
| 6 | 4 | 2 | 6 | 8 | 7 |
| 10 | 7 | 2 | 10 | 7 | 7 |
| 14 | 9 | 2 | 14 | 15 | 6 |
| 18 | 8 | 2 | 18 | 21 | 6 |
| $k$ in true models = 2 | | | $k$ in true models = 6 | | |

(c) **Dataset size = 1,000 sequences**.

Table 3.3: State spaces of HMM-As and HMM-CLs learned in simulations ($k$ denotes number of states).



(a) Number of states in true models = 2.



(b) Number of states in true models = 6.

Figure 3.8: HMM-As and HMM-CLs learned from simulated data: number of parameters for different cases.

## 3.6 EXPERIMENTS WITH REAL-WORLD DATASETS

In this section, we describe experiments for learning symmetric HMMs, HMM-CLs and HMM-As from real-world datasets originated from several domains. In order to empirically determine state spaces and assess the learned models, we used a procedure similar to the one described in Section 3.5. It differs in that real-life datasets are split in two parts: one for selecting models via 10-fold cross-validation (using 80% of the data), and the remaining portion for assessment of generalization.

### 3.6.1 *Datasets*

The datasets considered in this section are summarized in Table 3.4 and described next.

| **Dataset** | $n$ | **Description** | **Sequence data** |
|---|---|---|---|
| Volvo | 3 | Event logs of software incidents | 151 (50) |
| Rabobank | 6 | Event logs of software incidents | 500 (30) |
| Airquality | 12 | Urban pollution monitoring | 40 (48) |
| Printer $\mathcal{R}_1$ | 7 | Performance of printing nozzles and maintenance activity | 27 (15) |
| Printer $\mathcal{R}_2$ | 7 | Performance of printing nozzles and maintenance activity | 52 (15) |
| Printer $\mathcal{R}_3$ | 7 | Performance of printing nozzles and maintenance activity | 58 (15) |

Table 3.4: Summary of real-world datasets. The *sequence data* column shows the number of sequences together with sequence duration in parenthesis.

#### 3.6.1.1 *Business process data*

The business process dataset consists of event-log records on software incidents related to, e.g., software bugs, hardware problems, among others within the scope of ICT company departments. In general, these datasets are often used for process mining, covering tasks such as conformance checking (i.e. checking whether the business process specification complies with the running process), process discovery and process enhancement [1]. Learning business models as done in process mining field often intends to capture the underlying sequential behavior of actions within events. Thus, given a collection of events, business models are fitted to this data in order to represent different ways in which an event can develop over its lifetime.

As opposed to business models (e.g. workflow-like models), where one often wants to understand the internals of events, in this section we learn a complementary behavior from event-logs data in the form of influences among events. As this is less evident from data and involves multivariate observations (since events are typically composed of several features), this is a challenging problem, for which this section offers an HMM-based solution. We considered two datasets from the BPI (business process intelligence) challenge, described as follows.

VOLVO DATASET The Volvo IT Belgium dataset [166] consists of event logs of software incident registered during the period of 2011-2012. Each data point describes an incident by means of three features: incident impact, push to front (i.e. whether the incident was handled by a service desk team or required other specialized teams as well), and country (referring to whether the incident involved employees from different nationalities). The Volvo data was split in sequences such that each sequence has approximately 5 days of incidents.

RABOBANK DATASET The Rabobank Group ICT dataset [55] consists of event-log records of software incidents over the period of 2011-2014, however from a different software domain than the Volvo dataset. We considered the part of the data related to interactions, which registers the first contact between a user of a software component and a service desk team. An interaction call can lead to an incident or not. Each interaction is described in the Rabobank dataset by a set of six features: type of involved item (e.g. application, hardware, network-related issues), impact (in case of service disruption), priority, category (i.e. whether the event refers to a request for information or an incident), first call (i.e. whether the interaction could be solved by service desk team or led to an incident for further resolution), and handle time (i.e. the amount of time to resolve the service disruption).

Learning HMM-As for business processes aims in first place to provide well fitted models, but also aims to discover different dynamics that might govern the generation of incidents and interactions. This can then be turned into practical knowledge, e.g., to assist decision makers when devising more effective and resource-saving business processes. We shall discuss more on this in Section 3.6.3.

#### 3.6.1.2 *Airquality data*

The Airquality dataset contains data on gas pollutants in the context of urban pollution monitoring [172]. The feature set is composed by two different sources of information: a set of reference pollutant concentrations provided by conventional stations, and a set of measurements provided by a multi-sensor device. Originally, the Airquality dataset was used to evaluate and calibrate sensor devices for estimating the concentration of pollutants, as a technological means for low cost and convenient air monitoring across urban spaces. In the original paper [172], simple positive correlations among sensors data were found to influence the prediction accuracy, hence we provide a complementary analysis to how these correlations develop in a sequential way. We considered a feature set with 12 variables corresponding to the original measurements, which were discretized for the experiments in this section. The records for the variable for the ground-truth non-methane hydrocarbons were not considered, as they were absent in most of the cases.

#### 3.6.1.3 *Printers data*

We also considered data to support understanding the behavior of modern, complex engineered artifacts (also called cyber-physical systems), for which we use a large-scale printer as a case study. Whereas engineers understand the functioning of the individual components in considerable depth and detail, as a consequence of their intricate design they find it much more difficult to understand the behavior of the artifacts at a certain level of abstraction, as well as their interaction. In order to learn the temporal behavior of such systems, data was gathered from three printers of the same printer family, where the usage of the printers differs as function of time, and as function of the print jobs being rendered. In this case study, we focus in particular on one component – the nozzle – that aims at jetting ink on the paper. The behavior of nozzles as function of time depends on several factors, such as the quantity of ink used, time since last maintenance and some environmental parameters.

The logs that were considered consist of a 1-year record of nozzle-related factors continuously monitored. We considered a key maintenance action that is performed by the machine from time to time, and gathered data on nozzle-related components between each maintenance occurrence, such that each (multivariate) observation includes the following features: interval duration (i.e. the length of time since the previous maintenance action), total workload, frequency of another related maintenance action, and color-related features. The goal of our experiment is to discover relations between features and how it influences the proper functioning of the nozzles.

### 3.6.2 *Results*

We first report results on fit quality based on model selection, where Figure 3.9 shows the mean validation log-likelihoods in function of the number of states. These results show that the structural simplicity of independent HMMs could be compensated to some extent by learning larger state spaces, and thus model quality similar to that attained by more structured models (i.e. standard HMMs and HMM-As) could be achieved. However, this was not possible in all the cases, in particular in the business process datasets. In these cases, prior to achieving overfitting the structured models had a much better fit, suggesting that in some cases the presence of non-trivial structure over observables can be deemed crucial in order to obtain good models.

With respect to the structured models, contrasting standard HMMs with HMM-As indicates that HMM-As achieved superior fit on some cases (e.g. Airquality and Rabobank) and similar model quality on the remaining ones. The learned HMM-As better fitted the data than Chow-Liu HMMs in general as well. It is interesting to note that HMM-CLs impose tree structures to its emissions, something that might not be always beneficial. For example, in the Volvo and Airquality cases, the results suggest that even a symmetric model as the HMM-S was able to provide better results than HMM-CLs, which is interesting as the HMM-S does not necessarily learn a connected structure for the emissions space. In general, it can also be observed from Figure 3.9 that structured models as standard HMMs and Chow-Liu HMMs overfit much more easily than HMM-As, suggesting that HMM-As provide more parsimonious solutions to these real problems.

As Figure 3.9 shows, dynamic Bayesian networks (DBNs) were also learned from the real-world datasets, whose results indicate that DBNs provided consistently inferior model quality than HMMs. Although these results are not directly related to comparing HMMs, they suggest that modeling autoregressions alone (as in DBNs) is not a guarantee for good fit in real-life datasets: modeling multiple (and possibly structured) distributions via hidden states can be more powerful, yet no autoregressions are modeled by these HMMs. A question that could be of interest is whether including autoregressions in HMM-As would bring real benefits to such models.

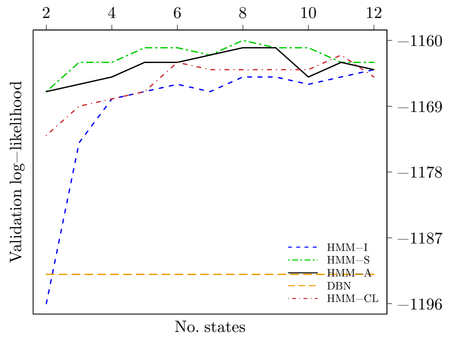
Having discussed the dynamics of model quality based on validation log-likelihoods experiments, we now use these results to select and learn models in order to discuss problem insight, as well as to assess their generalization. To this end, we select models in a flexible way: we pursue models with the highest fit, except when there are multiple models with similar quality, in which case we select the models with the lowest dimension. After finishing this, we learn models with the selected dimension using the entire datasets and measure their likelihood with testing data (i.e. data that was not used in cross-validation).

The models learned for generalization assessment are summarized in Table 3.5. As there is no ground-truth model for the real-world datasets, to facilitate comparison we used *normalized log-likelihoods* as follows:

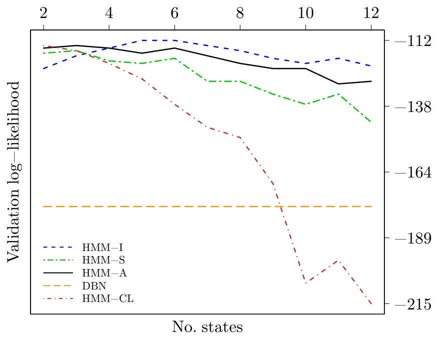
$$\mathrm{NLL} = \frac{-\log \mathcal{L}(\mathcal{B})}{c} \tag{3.21}$$

where $\log \mathcal{L}(\mathcal{B})$ is the log-likelihood of the model $\mathcal{B}$, and $c$ is a normalizing constant given by $c = mTn$, with $m$ being the number of sequences, $T$ the sequence length, and $n$ the number of features in the dataset.

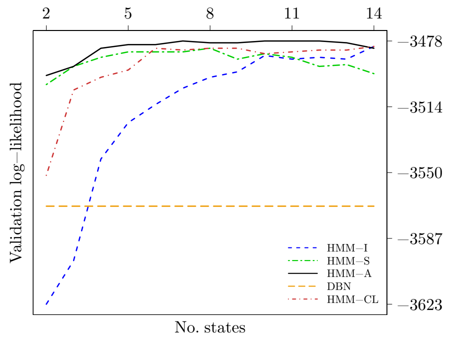
As Table 3.5 shows, HMM-As generalized consistently better than symmetric HMMs and Chow-Liu HMMs. We further computed 95% confidence intervals (CIs) for these models as shown in Table 3.6, in order to check for the robustness of the generalization assessment (intervals were obtained by means of bootstrapping the testing datasets for 2,000 times in each case). The CIs show that HMM-As could provide significantly better model quality on most scenarios of business process and Airquality cases. Significance in favor of HMM-As was also obtained in the printers cases, although HMM-As can be considered better than Chow-Liu HMMs in these cases but not significantly. This is explained by the fact that the HMM-A states are all virtually associated to forests that are sparser than trees, as in Printer $\mathcal{R}_3$ (Figure 3.14) and Printers $\mathcal{R}_1$ and $\mathcal{R}_2$ as well [23].



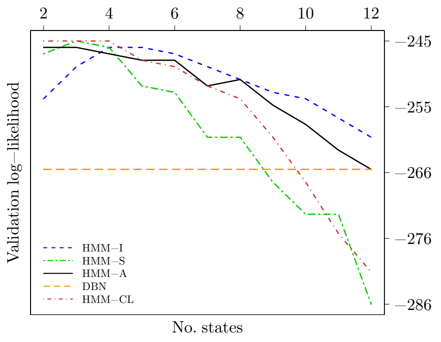
(a) Volvo



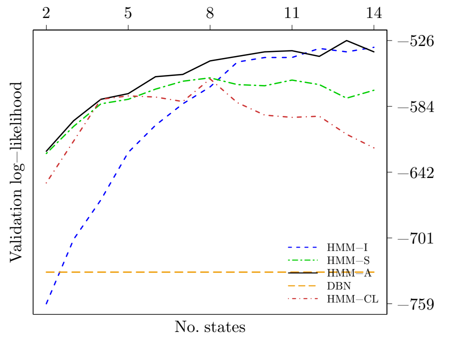
(b) Printer $\mathcal{R}_1$.



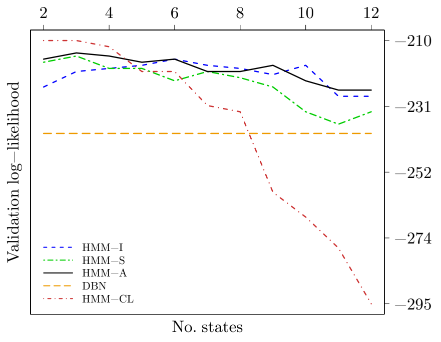
(c) Rabobank.



(d) Printer $\mathcal{R}_2$.



(e) Airquality.



(f) Printer $\mathcal{R}_3$.

Figure 3.9: Cross-validation log-likelihoods achieved by DBNs, symmetric, Chow-Liu, and asymmetric HMMs in real-world datasets. Each point represents the mean validation log-likelihood over 10 folds.

From the results on real-data discussed in this section, it seems fair to conclude that not only more structure is beneficial for HMMs to better capture real-life problems, but also the *right* additional structure as provided by HMM-As by their state-specific Bayesian-network distributions. The number of parameters in HMM-As was consistently lower than those of standard HMMs, independent

| | HMM-I | | | HMM-S | | | HMM-CL | | | HMM-A | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Dataset** | $k$ | **NLL** | **#Pa.** | $k$ | **NLL** | **#Pa.** | $k$ | **NLL** | **#Pa.** | $k$ | **NLL** | **#Pa.** |
| Volvo | 8 | 65.2 | 87 | 5 | **64.7*** | 59 | 6 | 65.2 | 65 | 5 | **64.7*** | 50 |
| Rabobank | 10 | **48.5*** | 159 | 5 | 48.7 | 214 | 6 | **48.5*** | 101 | 4 | **48.5*** | 73 |
| Airquality | 8 | 32.9 | 159 | 8 | 32.1 | 623 | 8 | 35.4 | 247 | 6 | **31.3*** | 170 |
| Printer $\mathcal{R}_1$ | 5 | 49.8 | 59 | 3 | 48.7 | 53 | 3 | 46.6 | 47 | 3 | **46.0*** | 36 |
| Printer $\mathcal{R}_2$ | 4 | 60.5 | 43 | 3 | 61.7 | 65 | 3 | 60.4 | 47 | 3 | **59.9*** | 43 |
| Printer $\mathcal{R}_3$ | 4 | 48.0 | 43 | 3 | 46.0 | 56 | 3 | 46.4 | 47 | 3 | **45.7*** | 36 |

Table 3.5: Generalization assessment of learned models on real-world datasets. Notation: $k$ denotes the number of states, NLL the normalized log-likelihood, and *#Pa.* the number of parameters. Results that generalized the best are bold-faced and followed by an asterisk.

| **Dataset** | **Asymmetric vs. Independent** | **Asymmetric vs. Standard** | **Asymmetric vs. Chow-Liu** |
|---|---|---|---|
| Volvo | **[-0.86, -0.11]**$^{**}$ | $[-0.14, 0.29]\dagger_S$ | **[-0.78, -0.21]**$^{**}$ |
| Rabobank | $[-0.15, 0.15]$ | $[-0.47, 0.06]*_A$ | $[-0.02, 0.26]\dagger_{CL}$ |
| Airquality | **[-3.17, -0.17]**$^{**}$ | $[-3.08, 0.61]*_A$ | **[-10.30, -1.77]**$^{**}$ |
| Printer $\mathcal{R}_1$ | $[-10.35, 0.46]*_A$ | $[-7.88, 0.37]*_A$ | $[-1.32, 0.05]*_A$ |
| Printer $\mathcal{R}_2$ | $[-1.53, 0.74]*_A$ | **[-4.16, -0.38]**$^{**}$ | $[-1.58, 1.11]*_A$ |
| Printer $\mathcal{R}_3$ | **[-4.79, -0.3]**$^{**}$ | $[-1.67, 0.26]*_A$ | $[-3.35, 3.48]$ |

Table 3.6: 95% bootstrap confidence intervals for the differences on generalization assessment (real-world datasets). Negative values indicate better fits for HMM-As. Notation: $**$ = HMM-A is significantly better; $*_A$ = HMM-A is better but not to a significant extent; $\dagger_X$ = model X is better but not to a significant extent.

HMMs and Chow-Liu HMMs, suggesting that diverse local structure exists which could be discovered by HMM-As.

### 3.6.3 *Problem insight*

We discuss in this section problem insight that can be gained from the learned HMM-As. We stress that from a fundamental perspective, where Bayesian networks are tools to facilitate reasoning with statistical independences, the fact that HMM-As can provide multiple graphical structures to explain how dynamic systems evolve over time (e.g. a business process) represents additional insight by its very nature. This contrasts to symmetric HMMs, where all those specificities are lost (or hidden across a number of CPTs at most), thus much less insight is likely to be gained.

#### 3.6.3.1 *Business process models*

Figure 3.10 shows the HMM-A learned from the Rabobank case (CPTs are not shown). The model shows that *Type* is unconditionally independent of *Impact* and *Priority* on the bottom right-most state, while this is not the case on the top right-most state. This structural information might be used, e.g., to further develop different policies for scheduling different types of interactions in different moments: if the system is assumed to be in the bottom right-most state, a more flexible scheduling might be possible, where different types of interactions do not need to be handled by priority or impact, but instead could be handled by the expected time to be solved (due to the relationship with *Handletime*). On the other hand, if the system is in the bottom left-most state, *Type* is still unconditionally independent of *Impact*, but its unconditional independence of *Priority* no longer holds: in fact, such state seems to act as a bridge for the two aforementioned states.

The aforementioned problem insight cannot be derived from the (almost fully connected) graphical structure of the learned HMM-S partially shown in Figure 3.11, nor from the learned HMM-I. At a higher level of abstraction, HMM-As also allow for new insight obtained by combining the local state properties with state-transition probabilities: this shows that batches made of few software-incident events that share independence properties are produced over time.

Figure 3.12 shows the HMM-A learned from the Volvo dataset. As in the Rabobank case, this HMM-A is made of different graphical structures that lead to different independence relations, whereas the standard HMM has a fully connected structure as shown in Figure 3.13. Finally, we note that in asymmetric models, not only probabilistic relationships change, but also the structure in each state, providing evidence that these models capture an additional facet of the different stages the underlying dynamic system can transit to.

#### 3.6.3.2 *Printers model*

Figure 3.14 shows the HMM-A learned from Printer $\mathcal{R}_3$ dataset (the other printer models were discussed elsewhere [23]). This model suggests that the behavior of such large-scale printer alternates between two modes in the long run, which can be distinguished based on how the color rates $C_1$, $C_2$, $C_3$, and $C_4$ interact with the other observables. For example, once the printer is assumed to be in the right-most top state, one could decide on whether the number of maintenance performed could be altered in order to save resources, as this variable will not affect the colors' performance. However, this is probably not the case for most of the colors if the printer is in the center state, where those colors do interact with other observables. The standard HMM learned for this printer is shown in Figure 3.15, lacking from such specific alternation behavior that could be discovered by means of the HMM-A, as the colors variables are connected to all the other observables (whether directly or not).

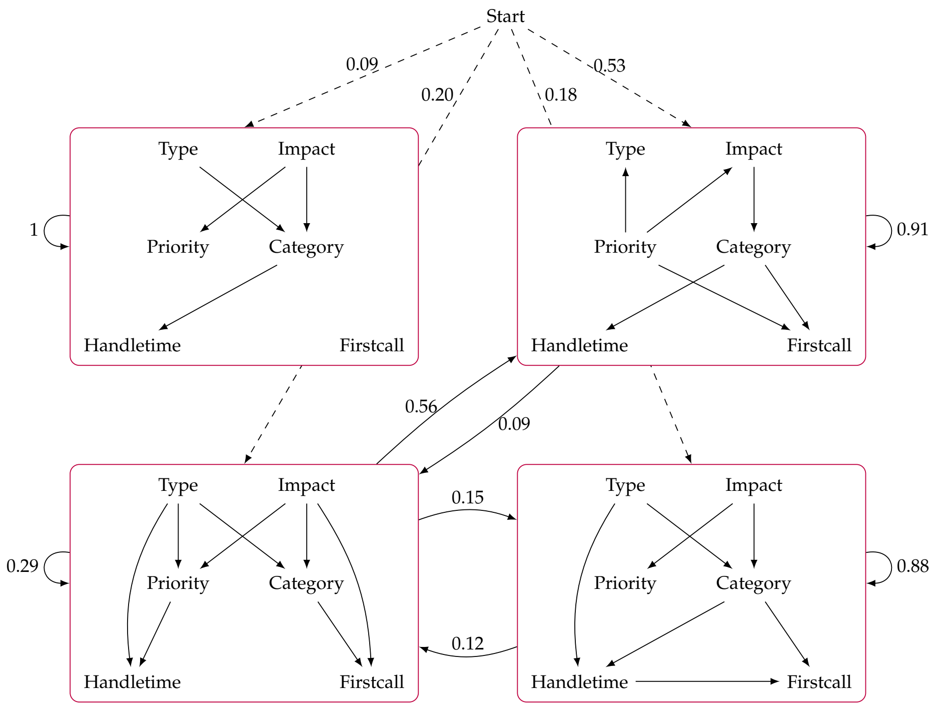


Figure 3.10: HMM-A learned from the Rabobank dataset. Dashed arcs indicate initial probabilities.

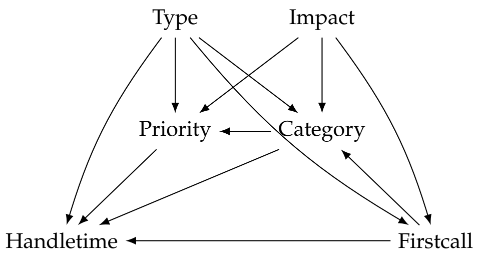


Figure 3.11: Graphical structure for emissions of the standard HMM learned from Rabobank dataset.

## 3.7 RELATED WORK

Analyses of the sensitivity of Bayesian networks to parameter change are relatively numerous [34, 72, 130], however that does not seem to be the case when it comes to the sensitivity to the graphical structure. There is some research on how model structure affects accuracy in medical diagnosis problems [131], where the authors have shown that the accuracy was not significantly sensitive for disturbances on model structure, considering certain medical cases and diagnostic criteria. In the

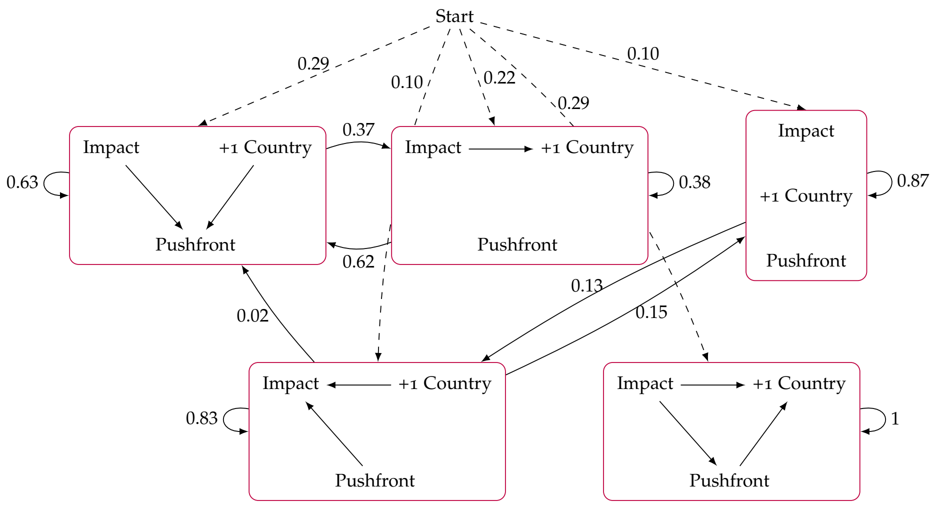


Figure 3.12: HMM-A learned from Volvo dataset.

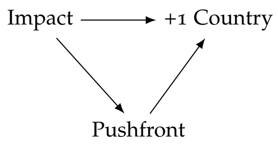


Figure 3.13: Graphical structure for emissions of the standard HMM learned from Volvo dataset.

context of HMMs, however, the results shown in this chapter suggest a different conclusion: with respect to model fit, modeling additional and specific (by means of distribution asymmetries) seemed very important for achieving better model quality. Nevertheless, these conclusions are not necessarily contradictory in principle, as the employed criteria differ and so do the type of models. As in HMMs the state space dimension is a rather important parameter, modeling non-trivial structure that can lead to smaller state space seems crucial to such models, while this might not be the case for some static Bayesian networks. In fact, it is a general belief in the Bayesian networks field that non-trivial structure matters for better handling real-world problems [51, 67, 150].

In this work, we attempted to provide a better understanding of the effects of modeling asymmetries on the feature space in HMMs, which is somewhat lacking in the literature. This involved a more thorough comparison of HMM-As with several families of HMMs: this included not only independent HMMs, but also standard HMMs and Chow-Liu HMMs, the latter being able to model simpler asymmetries than HMM-As. Finally, the chapter showed experiments involving models of DBNs, which are typically learned without hidden or latent variables.

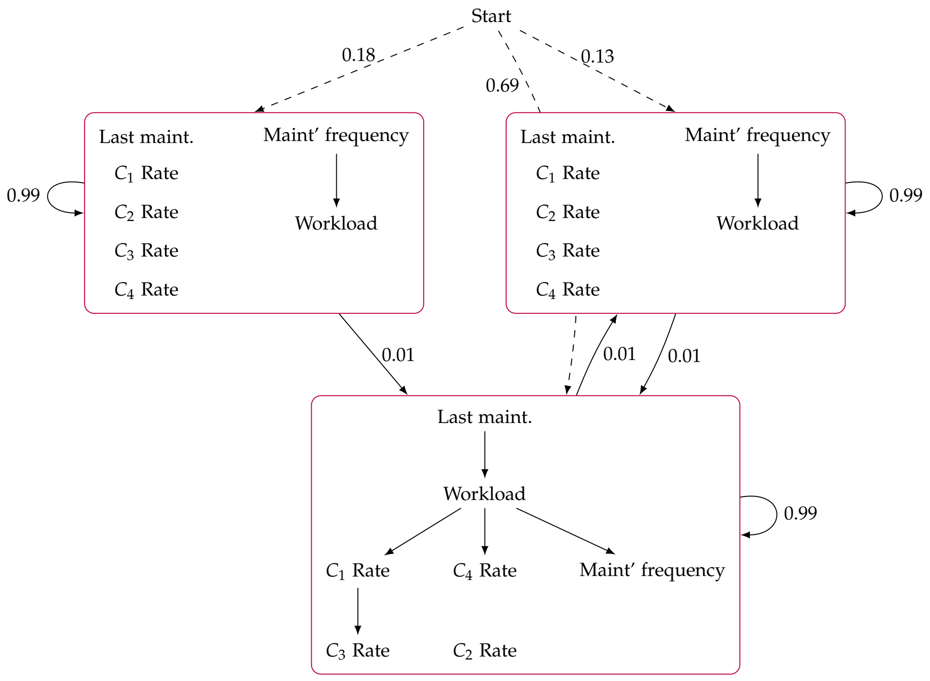


Figure 3.14: HMM-A learned from Printer $\mathcal{R}_3$ case.

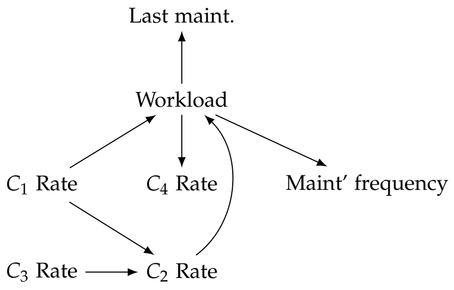


Figure 3.15: Graphical structure for emissions of the standard HMM learned from Printer $\mathcal{R}_3$ dataset.

## 3.8 CONCLUSIONS

In this chapter, we proposed a new family of HMMs called asymmetric hidden Markov models. HMM-As explicitly capture distribution asymmetries inherent to many real-world problems, by means of associating individual hidden states to arbitrary Bayesian networks. We showed that, in principle, symmetric HMMs (e.g. independent and standard HMMs) can have their state space or emissions structure arbitrarily extended for representing structured distributions.

Nevertheless, empirical results showed that this capability was not enough for guaranteeing comparable model quality, due to model overfitting (because of too many states or too dense emission structures). A similar conclusion holds for Chow-Liu HMMs, suggesting that going beyond tree-shaped asymmetries as done by HMM-As can be beneficial.

In some real-world cases, adding structure, either via symmetric or asymmetric models, allowed for relevant model quality improvement, while the simplest model (i.e. the independent HMM) was good enough in other cases. This model selection issue could be adequately addressed by HMM-As, which provide enough flexibility to reduce the need for selecting a particular HMM architecture a priori.

Computationally, learning HMM-As introduces an additional burden due to structure learning, compared to symmetric HMMs. Nevertheless, experiments indicated that good-quality HMM-As with compact state spaces could be obtained by using graphical structures found by common search heuristics. Hence, in practice learning HMM-As using structural EM resulted in fact in shorter running times compared to learning symmetric HMMs using standard EM in many cases. Furthermore, HMM-As learned from real-world datasets with varied sizes and number of observables were shown to bring additional problem insight that cannot be readily obtained from symmetric HMMs.

Several paths for further research can be considered. To some extent, HMM-As can be seen as tools for summarizing hidden Markov models with larger states spaces into models with more compact state spaces, as shown in the real-world experiments. We would like to further evaluate whether HMM-As act as *model summarizers* in more general settings, e.g. when the data generation mechanism has no explicit asymmetries (as in HMM-Is), and when it consists of different kinds of asymmetries (such as autoregressions, as in dynamic multinets).

It could be also of interest to exploit the sensitivity of differences between state-specific networks, e.g., along the lines of sensitivity analysis research. This could help, e.g., to eliminate too specific arcs that do not significantly contribute to model quality, thus allowing for more compact models. As we observed in real-world experiments and in simulations, several advantages obtained with HMM-As were more prominent when the problem had higher number of observables. Hence, we intend to further investigate such scaling aspect, as well as consider other real-world cases with more features and different types of observables (e.g. continuous and hybrid ones). Finally, we would like to compare the identification of asymmetries in sequential models as HMM-As with other approaches, such as knowledge compilation [39] and dynamic chain event graphs [5].

## 3.A PROOFS

*Proof of Proposition 3.1.* Let $\mathcal{M}$ be the given HMM-A and $\mathcal{M}'$ be a standard HMM, where both models are defined over $(\mathbf{X}, S)$. We construct $\mathcal{M}'$ for simulating $\mathcal{M}$ as follows. Let $G_F$ be directed acyclic graph over $\mathbf{X}$ that is also fully connected. Add an arc from $S$ to each $X_i \in \mathbf{X}$, and define the result as the graphical structure of the emissions of $\mathcal{M}'$. By the chain rule from probability theory, a fully connected structure can represent any probability distribution, hence, the distribution of each state in $\mathcal{M}$ can be represented by a state in $\mathcal{M}'$ by adequately parameterizing the CPTs on the emissions of $\mathcal{M}'$. This allows us to obtain $P'(\mathbf{X}^{(t)} \mid s^{(t)}) = P(\mathbf{X}^{(t)} \mid s^{(t)})$, for every state $s \in \text{dom}(S)$.

Finally, we set the initial and transition distributions of $\mathcal{M}'$ to the same as those from $\mathcal{M}$. Thus, we conclude that $P'(\mathbf{X}^{(0:T)}, S^{(0:T)}) = P(\mathbf{X}^{(0:T)}, S^{(0:T)})$. □

*Proof of Proposition 3.2.* We construct in the following an independent HMM $\mathcal{M}'$ for simulating a given asymmetric HMM $\mathcal{M}$ with $k$ states. The observables are assumed to follow Bernoulli distributions each (an extension to multinomial distribution is straightforward). We denote by $P$ and $P'$ the joint distributions over $(\mathbf{X}^{(0:T)}, S^{(0:T)})$ of $\mathcal{M}$ and $\mathcal{M}'$ respectively. Note that the state-specific BN associated to any state in $\mathcal{M}$ is a BN over $n$ variables, hence its joint distribution can be completely characterized with at most $2^n - 1$ independent parameters, where we denote by $\theta_\mathbf{x}$ the parameter associated to the assignment $\mathbf{x}$. For each $\theta_\mathbf{x}$ from state $s_i$, we define a state $s_{i\theta_\mathbf{x}}$ in $\mathcal{M}'$, as well as emission distributions of the form $P'(X_i = \top \mid s_{i\theta_\mathbf{x}}) \overset{\text{def}}{=} 1$ whenever $(X_i = \top)$ holds in $\mathbf{x}$. Following this procedure for all state-specific BNs from all states will result in $k2^n$ states in total in $\mathcal{M}'$. This finishes the construction of the emission distribution for $\mathcal{M}'$.

The remaining distributions of $\mathcal{M}'$ are constructed by scaling the corresponding distributions in $\mathcal{M}$ with the probability of each joint assignment of $\mathbf{X}$ as follows. For the initial distribution, we define

$$P'(s_{i\theta_\mathbf{x}}^{(0)}) \overset{\text{def}}{=} P(s_i^{(0)})P(\mathbf{x}^{(t)} \mid s_i^{(t)})$$

for each state $s_i$ from $\mathcal{M}$ and assignment $\mathbf{x}$. On the other hand, we define the transitions in $\mathcal{M}'$ as

$$P'(s_{j\theta_\mathbf{x}}^{(t+1)} \mid s_{i\theta_\mathbf{X}}^{(t)}) \overset{\text{def}}{=} P(s_j^{(t+1)} \mid s_i^{(t)})P(\mathbf{x}^{(t+1)} \mid s_j^{(t+1)})$$

where $s_{i\theta_\mathbf{X}}$ refers to any state originated from $s_i$. Here the instantiation of $\mathbf{X}$ in $\theta_\mathbf{X}$ is irrelevant: taking a transition from $s_i$ to $s_j$ is independent of the observation emitted by $s_i$, since $s_i$ is observed.

It is straightforward to verify that this construction produces a valid probability distribution, and it assures that $P'(\mathbf{X}^{(0:T)}) = P(\mathbf{X}^{(0:T)})$. As a side note, while $\text{dom}(\mathbf{X})$ does not change in the simulated model $\mathcal{M}'$, this is not the case for $\text{dom}(S)$, as opposed to the simulation of Proposition 3.1.

□

*Proof of Proposition 3.3.* The dynamic programming procedure for computing the expected statistics of one sequence stores the values of $\alpha$ in a $(T+1) \times k$ matrix, also known as lattice (or trellis) structure. The computation of a single $\alpha$ value has the cost of $\mathcal{O}(k+n)$ time as follows. In Equation 3.10, the summation amounts to $\mathcal{O}(k)$ as long as the transition distribution is encoded as a $k \times k$ matrix. The emissions in Equation 3.10, in turn, can be computed in $\mathcal{O}(n)$ time assuming each state-specific BN is conveniently encoded (e.g. using a graph traversal with look-up tables for the parameters), allowing one to compute the probability of any joint event in linear time. Thus, the total cost for each $\alpha$ value is $\mathcal{O}(n+k)$, hence the entire lattice for one sequence takes $\mathcal{O}(Tk(k+n))$ time. We build a lattice for $\beta$ as well, however the total cost per cell in this case changes to $\mathcal{O}(nk)$. Thus, the total cost for the lattice of $\beta$ is $\mathcal{O}(Tk^2n)$.

Once the lattices for $\alpha$ and for $\beta$ are done, we compute the expected statistic $\xi$. Based on Equation 3.9, computing one $\xi$ value amounts to $\mathcal{O}(k^2n)$, thus an entire $\xi$ lattice for one sequence takes $\mathcal{O}(Tk^3n)$ time. Finally, note that we can compute the expected statistic $\gamma$ by means of $\gamma_t(i) = \sum_{j=1}^{k} \xi_t(i,j)$, thus the lattice for $\gamma$ requires $\mathcal{O}(Tk^2)$ time by using the lattice of $\xi$.

The computation of all the lattices for an observation sequence is a sequential process in which the cost of $\xi$'s lattice dominates over the rest. Hence, the total cost of one E-step iteration for one sequence in HMM-As amounts to $\mathcal{O}(Tk^3n)$. $\square$