Max Markov Chain

Yu Zhang and **Mitchell Bucklew**
School of Computing and Augmented Intelligence, Arizona State University
{yzhan442, mbucklew}@asu.edu

Abstract

In this paper, we introduce Max Markov Chain (MMC), a novel model for sequential data with sparse correlations among the state variables. It may also be viewed as a special class of approximate models for High-order Markov Chains (HMCs). MMC is desirable for domains where the sparse correlations are long-term and vary in their temporal stretches. Although generally intractable, parameter optimization for MMC can be solved analytically. However, based on this result, we derive an approximate solution that is highly efficient empirically. When compared with HMC and approximate HMC models, MMC combines better sample efficiency, model parsimony, and an outstanding computational advantage. Such a quality allows MMC to scale to large domains where the competing models would struggle to perform. We compare MMC with several baselines with synthetic and real-world datasets to demonstrate MMC as a valuable alternative for stochastic modeling.

1 Introduction

Markov Chain (MC) is a simple but powerful tool for stochastic modeling. In a Markov chain, the current state variable is assumed to be conditionally independent of all ancestral state variables given a fixed number (a.k.a. the order of the chain) of its immediate parental and ancestral state variables (referred to as the *lags*). The simplicity of MC makes it a desired choice for modeling various stochastic processes [Cowles and Carlin, 1996; Elfeki and Dekking, 2001; Ye *et al.*, 2004; Pentland and Liu, 1999]. For domains with long-term dependencies, High-order Markov Chains (HMCs) are often needed. However, HMCs are generally expensive to maintain due to their space complexity exponential in the order of the chain, which also makes learning sample inefficient. Traditional approaches to addressing this issue, such as approximate HMCs (e.g., [Raftery and Tavaré, 1994; Berchtold, 1995; Berchtold and Raftery, 2002]) and variable-order Markov models (VOMMs) (e.g., [Roucos *et al.*, 1982]), suffer from excessive computational needs for parameter optimization, making them impractical for many real-world applications.

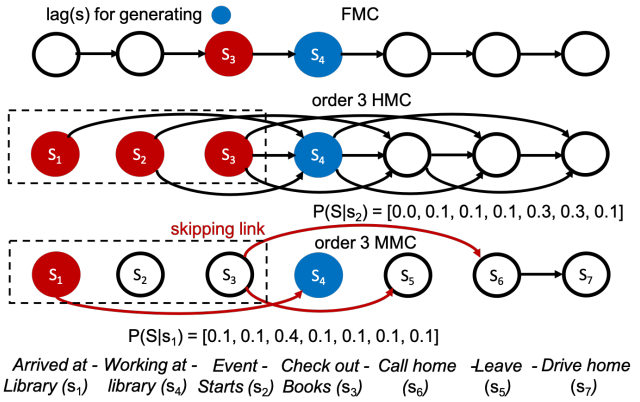

Figure 1: Graphical representations of, from top to bottom, First-order Markov Chain (FMC), High-order Markov Chain (HMC), and Max Markov Chain (MMC) specifications of the library example. The order is 3 for HMC and MMC. The size of the state space S is 7. In MMC, the transition matrix is parameterized by $P(S|S)$ (see examples above), which has the same space requirement as FMC. HMC would require $P(S|S,S,S)$. The subscripts of state variables (upper-case S) index into time steps and those of states (lower-case s) index into the state space.

Our aim is a class of models that are more restricted than HMCs in the types of data they can represent but otherwise much more efficient. Even though approximate HMCs and VOMMs are not designed to be limited in the same way, to achieve parsimony, they often consider the influences from the lags or subsets of lags independently for generating the next state. Such an assumption results in undesired modeling biases that restrict their data representation power. From this perspective, MMC may also be viewed as a special class of approximate models for HMCs (more discussion in Sec. 3.2).

In this paper, we introduce Max Markov Chain (MMC) where sparse and long-term correlations among the state variables are present. Our focus is on discrete time-homogeneous Markov models [Parzen, 1999] where the transition matrix, also referred to as the *state-generating* matrix in this work, does not change over time. See Fig. 1 for an illustration of the differences between MMC, First-order MC (FMC), and HMC. The first model assumption of MMC is that the next state is "generated probabilistically" by *one and only one* previous lag within the order of the MMC, resulting in *skipping links*. It is referred to as a max model since the lags for state

generation are assumed to compete with each other in a *context dependent* way: only the lag with the maximum generation probability is assumed to win, e.g., the red state variable ($S_1 = s_1$) for generating the state value of the blue state variable (S_4) in MMC. This is because, in Fig. 1, the maximum generation probability for s_1 is 0.4 given $P(S|s_1)$, and for s_2 is 0.3 given $P(S|s_2)$. Based on the model assumptions of MMC, the generation priority of state s_1 is higher than that of s_2, which is why the state variable $S_1 = s_1$ (instead of $S_3 = s_2$) generated the state value of S_4. Note that we presume in this scenario that both states s_1 and s_2 have a higher generation priority than s_4 ($S_2 = s_4$). These model assumptions for MMC simplify the structure of the chain and reduce overfitting (see Sec. 3.2).

One real-world domain where the MMC model assumptions are likely to hold is human behavior modeling where human behaviors are described as both reactive and deliberative [Schmidt, 2000]. Consider the following scenario: John arrives at the library and starts working on his class assignment in the library. However, an event kicks off in the library disrupting John's work and compelling him to leave. Before John leaves, he checks out a few literature books he plans to read at home. Since John would be arriving home earlier today, he makes a quick call before leaving campus and driving home. The sparse correlations (i.e., causal relationships) among the state variables in this scenario are shown in Fig. 1, which make it suitable for MMC. Modeling similar scenarios with HMC is possible but unnecessary while FMC would be insufficient given the long-term dependencies. Note that the "skipping links" in MMC differ fundamentally from the notions of jumping chains [Metzner *et al.*, 2009] and options [Sutton *et al.*, 1999], which are about the duration of transitions.

1.1 Contributions

We formally introduce MMC and provide an analytical solution for parameter estimation based on maximum likelihood. However, such a solution is generally intractable unless the state space is small. Based on the analytical solution, we provide a highly efficient greedy method that performs well empirically. Applying this method to both synthetic and real-world datasets, we compare MMC with several baselines in terms of prediction accuracy, sample efficiency, and computation time. Results confirm MMC as a valuable alternative for stochastic modeling.

2 Related Work

Markov chains (MCs) are introduced for stochastic modeling and have been applied in physics [Randall, 2006], computer science [Stewart, 1994], geography [Chin, 1977], behavior and social sciences [Benjamin, 1979]. MCs rely on the Markov assumption to simplify modeling, learning, and inference. They have also been generalized for more expressiveness and scalability, such as to consider hidden state [Baum and Petrie, 1966; Fine *et al.*, 1998] and factored models [Kearns and Koller, 1999]. However, MCs require an exponential number of parameters in the order of the chain, which makes them intractable for complex domains and causes learning to be sample inefficient. Approximate HMCs that are parsimonious and fast to learn are desired. Popular approximate models [Jacobs and Lewis, 1978; Raftery, 1985] use auxiliary variables to combine the influences from each of the lags individually for generating the next state. Such models have also been extended to consider mixtures [Berchtold and Raftery, 2002] where the influences to be combined can be specified with respect to more than one lag, leading to better approximations. However, there are two main limitations of the existing approximate HMC models. While they are more parsimonious than HMCs, parameter optimization is computationally challenging, often through complex numerical procedures [Raftery, 1985; Berchtold and Raftery, 2002]. Second, the influences for generating the next state are *not* context dependent, which limits the model flexibility and negatively impacts its performance.

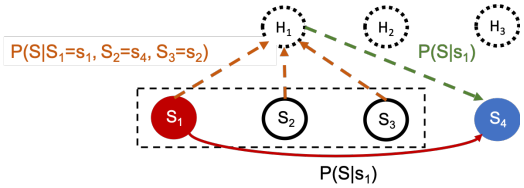

Figure 2: The first 4 state variables of the order 3 MMC in Fig. 1, represented by a Bayesian Network that replaces the associated skipping link (red) with the dashed links and an additional hidden variable (i.e., H_1). Later skipping links can be replaced in a similar fashion. The result is a Hierarchical Bayesian Network. MMC implicitly expresses such a structure (albeit being more restrictive in its parameter specification for model parsimony and efficiency) by introducing skipping links.

Variable-order Markov models (VOMMs) [Roucos *et al.*, 1982; Begleiter *et al.*, 2004] address the second issue of approximate HMCs by learning variable orders (that are context dependent) from data, which also contributes to parameter reduction. Summary Markov Models (SuMMs) [Bhattacharjya *et al.*, 2022] generalize VOMMs by restricting history information to influencing sets. Unfortunately, parameter estimation for these models is still computationally expensive. Neither VOMMs or SuMMs consider skipping links. MMC uses these links to simplify the model structure and impose model sparsity while building more flexibility into its temporal dynamics (see Fig. 1). The simpler model structure contributes to a highly efficient parameter estimation method for MMC. MMC can be further generalized to sequential data with denser correlations, resulting in a spectrum of models with increasing expressiveness that converge to HMCs.

The model structure of MMC may appear similar to skip-chain sequence models [Galley, 2006; Sutton and McCallum, 2004] where the skipping structure is known a priori (unlike with MMC). The closest work that learns skipping links is VDJ-HMM [Petropoulos *et al.*, 2017]. However, MMC is for time-homogeneous MCs while VDJ-HMM is for non-homogeneous MCs, so the temporal dynamics (TD) of the models are different. The TD in VDJ-HMM follows a separate FMC stochastically while the TD in MMC is fixed given the context (i.e, lags).

MMC is fundamentally different from jumping chains and options [Metzner *et al.*, 2009; Sutton *et al.*, 1999]; it models varying temporal stretches of skipping links instead of vary-

ing transition duration. MMC may be converted to a Hierarchical Bayesian Network (HBN) [Gyftodimos and Flach, 2002] with a hidden middle layer for deciding which state variable in the lags dominates the generation of the next state (see Fig. 2). However, representing an MMC as an HBN would no longer be parsimonious. MMC may also be viewed as a special type of Contingent Bayesian Network (CBN) [Milch *et al.*, 2005] where the network structure is context dependent: the links are labeled with conditions to determine when they are active. We will show that parameter estimation for MMC can be done efficiently while the estimation for CBN, in general, is challenging.

3 Approach

Next, we first provide the background for High-order Markov Chain (HMC) and a popular approximate model for HMC. We then introduce Max Markov Chain and discuss parameter optimization (learning) with data. We focus on complete data and defer learning with partial data to future work.

3.1 Preliminaries

A discrete time-homogeneous Markov chain of order K is specified by the probability distributions, captured by the transition or state-generating matrix, of the next state given the previous K lags:

$$P(S_{t+K}|S_{t:t+K-1}) \tag{1}$$

The parameter size of this model is $O(M^{K+1})$, where M is the size of the state space. MTD [Raftery, 1985], an approximate HMC model, approximates Eq. (1) as follows:

$$P(S_{t+K}|S_{t:t+K-1}) = \Sigma_l \lambda_l q_{(t+l)(t+K)} \tag{2}$$

where $q_{(t+l)(t+K)}$ is an element in an $M \times M$ transition matrix Q, capturing the influence from state S_{t+l} to S_{t+K} individually; λ_l is the weight associated with lag l and satisfies $\sum_l \lambda_l = 1$. MTD has a size of $O(M^2 + K)$. Even though MTD is more parsimonious than HMC, it makes the assumption that the influence is *context independent*: the influence from one lag is independent of the other lags. Also, parameter estimation with MTD is computationally challenging.

MMC addresses the issues of MTD by simplifying the model structure while allowing skipping links for more flexible temporal dynamics. In particular, MMC makes the assumption of sparse correlations such that only one of the lags generates the next state (i.e., the state value of S_{t+K}). However, the determination of such a lag is *context dependent* in MMC to allow the influences from the other lags to be passed implicitly. Model learning for MMC may be viewed as discovering causal relationships [Pearl, 2003] and hence is sample efficient. While the MMC model assumptions are restrictive, we will show in our evaluation that MMC performs well with real-world datasets, suggesting that domains with sparse correlations may be common.

3.2 Max Markov Chain

Our innovation is Max Markov Chain (MMC) for domains with sparse and long-term correlations. An MMC is specified as follows:

$$P(S_{t+K}|S_{t:t+K-1}) = P(S_{t+K}|S_{t+l^*}) \tag{3}$$

where $l^* = argmax_{S_{t+K},l} P(S_{t+K}|S_{t+l})$.

Intuitively, the generation of the next state is dominated by one of the lags with the maximum state-generation probability, or *generation probability* for short. Correspondingly, the state-generation (or transition) distribution is referred to as the *generation distribution*. Essentially, the maximum generation probability for each lag determines its *priority for state generation*, and the generation distribution of the next state depends only on the lag with the highest priority. Theoretically, such a specification still represents an order K Markov chain since the generation distribution of S_{t+K} depends on all the lags, albeit in a restricted way (i.e., winner-take-all). Even though MMC only uses the same amount of parameters as a First-order Markov Chain (FMC) (i.e., $O(M^2)$), such a formulation allows it to encode varying long-term dependencies via skipping links. The price to pay is that MMC is expected to have difficulty when the link structure (i.e., temporal stretches) is fixed in the data, such as when the data is generated by FMCs or skip chains. From this aspect, MMC may be viewed as a special class of approximate HMCs, since the traditional approximate HMC models (e.g., MTD) are strict generalization of FMCs and skip chains. However, MMC can be extended to address this limitation by augmenting to include links with fixed temporal stretches at the cost of model parsimony.

We could have defined a simpler MMC model by specifying the transition or state-generation matrix of $P(S|S)$ independently from the priorities of the states. In MMC, however, we relate these priorities to the maximum generation probabilities (MGPs) to introduce a set of constraints on the generation distributions (specified in the matrix), which represents a way to reduce overfitting: it prevents assigning high MGPs to the states with low priorities. Intuitively, states with lower priorities tend to generate fewer states and are hence more subject to overfitting. MMC alleviates overfitting by constraining the generation distributions so that the model generally associates higher uncertainty with states having lower priorities. For example, if a state is unseen during training to have generated another state that however appears during testing, it would likely result in a low data likelihood for the testing data under the simpler MMC model, which is undesirable.

Proposition 1. *Every MMC has a well-defined probability distribution.*

This is easy to see since every MMC has an associated HMC of the same order. It can also be derived from results in [Milch *et al.*, 2005] when viewing MMCs as HMCs with contingent edges, given that HMCs do not contain loops.

3.3 Parameter Estimation

Given training data $\mathcal{D}$ with each sample in the form of $d : s_{t:t+K-1} \rightarrow s_{t+K}$ $(d \in \mathcal{D})$, the problem can be described as maximizing the data likelihood under the i.i.d. assumption:

$$\max_{\mathcal{M}} P(\mathcal{D}|\mathcal{M}) = \max_{\mathcal{M}} \Pi_d P(d|\mathcal{M}) \tag{4}$$

where $\mathcal{M}$ denotes the space of MMCs. Directly optimizing Eq. (4) is infeasible given the infinite MMCs. To discuss our solution, first, we introduce *State Generation Order* (SGO):

$\rightarrow$	s_1	s_2	s_3
s_1	0	2	2
s_2	2	0	0
s_3	1	0	4

Table 1: A data generation table for an MMC of order 2 with 3 states based on the training data 3333331231231 (note that state indices are used here to denote states for clarity). Assume the SGO given is $s_1 \succ s_2 \succ s_3$. For every two consecutive states (i.e., lags), the generation relationship for the next state is determined by which state among the two has a higher priority in the SGO. Each row is for a generating state (the first column) and the numbers in the row are the counts of a particular state generated by that generating state.

Definition 1 (State Generation Order (SGO))**.** *A state generation order specifies the order of priorities over the state space for state generation.*

We use $s_a \succ s_b$ to denote that state s_a has a higher priority than s_b for state generation. This requires $p^*_{s_a} \geq p^*_{s_b}$ to hold where $p^*_{s_a}$ ($p^*_{s_b}$) represents the maximum generation probability of s_a (s_b) for generating any state according to its generation distribution. It is clear that an MMC *introduces* a unique SGO, when assuming that ties are broken consistently. We thus can optimize Eq. (4) according to a 2-step process. In the first step, we determine the SGO of the state space. In the second step, we optimize the parameters according to the SGO identified. We will show next that parameter optimization under a given SGO is not difficult. In such a situation, optimizing Eq. (4) boils down to iterating through the set of SGOs, which is finite, to identify the SGO *after* parameter optimization that maximizes Eq. (4).

Parameter Optimization under a Specified SGO

Denote the space of SGOs as $\mathcal{O}$. The problem now becomes to consider optimizing for $P(\mathcal{D}|\mathcal{M}_o)$, where $o \in \mathcal{O}$ is a given SGO and $\mathcal{M}_o$ denotes the set of MMCs introducing the given SGO. Under the assumption that the data generation process is an MMC, with the given SGO, the generation relationship among the states in the data is also known (i.e., which lag among $s_{t:t+K-1}$ generated s_{t+K}). Refer to Tab. 1 for an example of how the generation relationship is determined. We denote the part of the data (i.e., a set of states) that is generated by state s in the training data $\mathcal{D}$ as $\mathcal{D}_s$. Each row in Tab. 1 corresponds to a $\mathcal{D}_s$ for the corresponding generating state (s) shown in the first column. Then, given an MMC $\mathcal{M} \in \mathcal{M}_o$, the contribution of $\mathcal{D}_s$ to the data likelihood is:

$$P(\mathcal{D}_s|\mathcal{M}) = p_1^{n_1} p_2^{n_2} ... p_m^{n_m} \tag{5}$$

where p_i is the generation probability of state s_i for the generating state s and n_i is the number of times s_i appearing in $\mathcal{D}_s$. m indexes into the state space. $p_1 : p_m$ jointly specify the generation distribution for s and are the parameters to learn. Note that we intentionally refrain from specifying to which state these values ($n_1 : n_m$ and $p_1 : p_m$) belong in Eq. (5) to avoid cluttering the notation: each generating state is associated with a separate set of these values (see Tab. 1 for $n_1 : n_m$ for the three different states). Next, take the log to transform Eq. (4) and express it using Eq. (5):

$$\max_{\mathcal{M}} \log P(\mathcal{D}|\mathcal{M}) = \max_{\mathcal{M}} \Sigma_s \log P(\mathcal{D}_s|\mathcal{M}) \tag{6}$$

Given that $p_1 : p_m$ specify a distribution, $P(\mathcal{D}_s|\mathcal{M})$ in Eq. (5) takes the unique maximum value when the p values are *aligned with the data*, such that $p_i = \frac{n_i}{\sum_m n_m}$ (referred to as the *optimal value setting*). The minimum value is at when one or more of the p values are 0. Furthermore, $P(\mathcal{D}_s|\mathcal{M})$ monotonically decreases as the p values deviate from the optimal value setting. Next, we introduce results that inform the parameter optimization process.

Lemma 1. *Given an SGO respected by the data, meaning that the priorities of the generating states according to the generation distributions estimated from data, i.e., $p_i = \frac{n_i}{\sum_m n_m}$, align with the SGO, the maximum data likelihood is achieved by setting the generation distribution according to $p_i = \frac{n_i}{\sum_m n_m}$ for each generating state, respectively.*

Proof. The proof is straightforward given that $P(\mathcal{D}|\mathcal{M}) = \Pi_s P(\mathcal{D}_s|\mathcal{M})$. Since $\{\mathcal{D}_s\}$ are fully specified by the SGO, under the given assumption above, the generation distribution for each state can be set according to the data (i.e., $p_i = \frac{n_i}{\sum_m n_m}$) without violating the constraints of the given SGO (i.e., $s_a \succ s_b$ requires $p^*_{s_a} \geq p^*_{s_b}$). Since it is also the maximum value possible under each $\mathcal{D}_s$, $P(\mathcal{D}|\mathcal{M})$ must also be the maximum for $\mathcal{D}$ under the SGO. □

In Tab. 1, assuming no violation with the SGO constraints, the generation distributions (i.e., p values) for s_1 and s_2 would be set to $\{0, 0.5, 0.5\}$ and $\{1.0, 0.0, 0.0\}$, respectively.

However, setting the MMC parameters this way for the example in Tab. 1 would violate the constraints of the given SGO $s_1 \succ s_2 \succ s_3$, since $p^*_{s_1} = 0.5 < p^*_{s_2} = 1.0$. To address the issue, given that the priorities of the generating states in the SGO are determined by their maximum generation probabilities, we can focus on these probabilities only to address any violations. First, we note that it is straightforward to determine the state that should be assigned the maximum generation probability for each generating state: the state being generated with the largest count, e.g., s_3 or s_2 for s_1, s_1 for s_2, and s_3 for s_3 in Tab. 1.

Lemma 2. *Given an SGO, assume that the maximum generation probabilities for all generating states are given (denoted by $\{p^*\}$) and they align with the SGO. For each generating state, respectively, the maximum data likelihood is achieved when the probabilities for the remaining states being generated are set according to $\min(p^*, \frac{n_j}{\sum_{m_k} n_{m_k}}(1 - \Sigma_k p_k))$ iteratively, with these states considered in the decreasing order of $\frac{n_i}{\sum_m n_m}$. k (j) indexes into such states that have (have not) been assigned a probability in the current iterative step; $\{n_{m_k}\}$ are counts for the states not yet assigned. States being generated with 0 counts will be assigned equal probability masses if any mass is left unassigned.*

Proof. When the maximum generation probability is given for a generating state s, the remaining states in the generation distribution of s contribute to the data likelihood via an equation that is similar to Eq. (5), except that the sum of the remaining p values would be $1 - p^*$. Given the monotonicity of Eq. (5) from the optimal value setting, we would need each remaining p value to be as close as possible to

$\frac{n_j}{\sum_{m_k} n_{m_k}}(1-p^*)$. When all such values are equal to or smaller than p^*, we would be done since the iterative steps above would be equivalent to setting the remaining p values according to $\frac{n_j}{\sum_{m_k} n_{m_k}}(1-p^*)$.

Otherwise, first, note that $\frac{n_j}{\sum_{m_k} n_{m_k}}(1-p^*)$ is the optimal value setting and we should deviate from it as little as possible. Given that the probability for any state being generated cannot exceed p^*, setting the values according to $\min(p^*, \frac{n_j}{\sum_{m_k} n_{m_k}}(1-\Sigma_k p_k))$ iteratively, ordered by $\frac{n_i}{\sum_m n_m}$ (equivalent to ordering by $\frac{n_j}{\sum_{m_k} n_{m_k}}(1-p^*)$), would serve the purpose. Setting the p values any other way would be suboptimal since we can always adjust them to be closer to the optimal value setting to improve the data likelihood. □

For the example in Tab. 1, consider a case when the maximum generation probability for the generating state s_1 is assigned to s_3 with a value of 0.4. In such a case, we cannot assign probabilities to the remaining states for s_1 according to $\frac{n_j}{\sum_{m_k} n_{m_k}}(1-p^*)$ since that would assign all the remaining probability mass (0.6) to s_2, leading to a conflict since s_1 is supposed to receive the maximum generation probability. In such a case, we assign the maximum possible to s_2 (0.4) and the remaining (0.2) to s_1. Note also that assigning s_2 to any value smaller than 0.4 would decrease the data likelihood.

The remaining challenge is to determine the maximum generation probability for each generating state. One may be tempted to assign the probability for each state according to $\max \frac{n_i}{\sum_m n_m}$ based on the data. However, problems can occur when the data does not align with the given SGO. Without the loss of generality, consider two different states s_x and s_y such that $s_x \succ s_y$ in the given SGO but $p^{\mathcal{D}}_{x^*} < p^{\mathcal{D}}_{y^*}$, where $p^{\mathcal{D}}_{x^*}$ $(= \frac{n^*}{\sum_m n_m})$ represents the maximum generation probability for state s_x estimated from data under the SGO and n^* denotes the largest count of the state being generated by s_x.

Definition 2 (Misalignment)**.** *A misalignment occurs when two states s_x and s_y satisfy that $s_x \succ s_y$ in the given SGO but $p^{\mathcal{D}}_{x^*} < p^{\mathcal{D}}_{y^*}$ based on the data.*

In Tab. 1, we can see that there is a misalignment between s_1 and s_2 ($p^{\mathcal{D}}_{s_1{}^*} = 2/4$ and $p^{\mathcal{D}}_{s_2{}^*} = 2/2$), and a second misalignment between s_1 and s_3 ($p^{\mathcal{D}}_{s_1{}^*} = 2/4$ and $p^{\mathcal{D}}_{s_3{}^*} = 4/5$), with the given SGO $s_1 \succ s_2 \succ s_3$. Misalignment is directionless so can be represented as an undirected edge between two generating states. In such a representation, all misalignments introduce a connected graph among the states. We refer to such a graph as a *misalignment graph*.

Lemma 3. *Given an SGO with a misalignment between s_x and s_y ($s_x \succ s_y$ in the SGO), the maximum data likelihood is achieved at $p^*_{s_x} = p^*_{s_y}$, where $p^*_{s_x}$ ($p^*_{s_y}$) represents the maximum generation probability for state s_x (s_y).*

Proof. Given $s_x \succ s_y$ in the SGO, it requires that $p^*_{s_x} \geq p^*_{s_y}$. Given Lemma 1, the maximum data likelihood is achieved at $p^*_{s_x} = p^{\mathcal{D}}_{x^*}$ and $p^*_{s_y} = p^{\mathcal{D}}_{y^*}$, respectively, without any SGO constraints. However, this would imply that $p^*_{s_x} < p^*_{s_y}$, leading to a violation of the SGO constraint. Given that the contribution to the likelihood is decreasing from its maximum as the p values deviate from the optimal value setting for both s_x and s_y, we show next that the maximum likelihood must be achieved at $p^*_{s_x} = p^*_{s_y}$ under the constraint of $p^*_{s_x} \geq p^*_{s_y}$.

We prove this result by contradiction. Assume that the maximum is achieved at $p^*_{s_x} > p^*_{s_y}$ instead. In such a case, we can update $p^*_{s_x}$ and $p^*_{s_y}$ to be the same value between $(p^*_{s_y}, p^*_{s_x})$ to move the p values of both states closer to their optimal value settings. This will increase the likelihood, resulting in a contradiction with the assumption made. □

Corollary 1. *For all states that are connected (directly or indirectly) via misalignments in the misalignment graph, the maximum data likelihood is achieved when their maximum generation probabilities are set to be the same.*

This is a direct result of Lemma 3. In other words, if s_x and s_y, and s_y and s_z, are misaligned, they must all have the same maximum generation probability to maximize the data likelihood. The implication here is that all the generating states will be divided into connected subgraphs and states in each connected subgraph must share the same maximum generation probability. In Tab. 1, given the misalignments, all the states must share the same maximum generation probability under the given SGO. In such a case, we can solve for this maximum generation probability by optimizing:

$$\max_p (1-p)^2 p^2 p^2 (1-p) p^4 \tag{7}$$

which reaches the maximum value at $p = 8/11$.

Theorem 1. *A local maximum of the data likelihood is achieved by assigning the same maximum generation probabilities to the generating states in a subgraph according to $\frac{\sum_c n^*}{\sum_c \sum_m n_m}$, where c denotes the states in a connected subgraph in the misalignment graph and n^* denotes the largest count of the state generated by each generating state in the subgraph. The remaining generating probabilities for each generating state are assigned according to Lemma 2.*

Proof. The contribution to the data likelihood of a connected subgraph assumes a similar form as Eq. (5), except that it involves multiple generation distributions that share the same maximum generation probability given Corollary 1. We can then compute the derivative of the likelihood (similar to that in Eq. (7)) to verify the result. When such a local maximum is also a global maximum will be studied in future work. □

This theoretical result provides an analytical solution for the parameter estimation problem under a given SGO. For the example in Tab. 1, we can use this result to derive the maximum generation probability for all the three states, $p_c = \frac{2+2+4}{2+2+2+1+4}$, given that they form a connected subgraph. This is consistent with the result derived in Eq. (7).

Parameter Optimization under Unknown SGO

Next, we extend parameter optimization to unknown SGO. Simply put, it involves iterating through all possible SGOs.

Theorem 2. *The MMC, resulted based on the best SGO $o^* \in \mathcal{O}$ after parameter optimization according to Theorem 1, is the MMC that maximizes the data likelihood among all possible MMC models.*

This is a direct result of Theorem 1. Since the process for determining the optimal set of parameters given the SGO requires only going through the data once, the computational complexity of the learning algorithm is $O(|S|!|\mathcal{D}|)$. The complexity is factorial in $|S|$ but only linearly in $\mathcal{D}$. Also, note that the complexity is indifferent to the order of the MMC, which allows the optimal solution to be computed for MMCs with small state spaces. The procedure for MMC parameter optimization is summarized below:

1. For each possible $o \in \mathcal{O}$
   - Estimate the generation probabilities from data with o
   - Determine the misalignments and subgraphs
   - Optimize the parameters according to Theorem 1
   - Compute $P(\mathcal{D}|\mathcal{M}_o)$ with the optimized parameters
2. Return the SGO achieving the maximum $P(\mathcal{D}|\mathcal{M}_o)$ along with the optimized parameters

Greedy Heuristic

The solution above, however, is intractable since the number of SGOs can be large when the state space is large. However, based on the theoretical result, we can derive efficient heuristics. A simple heuristic is considered that orders the states based on their maximum generation probabilities estimated from the data. At any step, we consider the remaining states as possible candidates for the state of the next highest priority in the SGO. This step requires going through the data once while ignoring part of the data that is generated by the states of higher priorities (i.e., considered in the previous steps). The state with the maximum generation probability among the remaining states is chosen to be the next state in the order. Such a heuristic results in an approximate method that runs in $|\mathcal{D}||S|$, allowing MMC to handle large domains. We observed in our evaluation that this simple method is highly efficient. We defer its theoretical analysis to future work.

4 Evaluation

For the purpose of comparison, we chose the High-order Markov Chain (HMC), a popular approximate HMC model (MTD [Raftery, 1985]), and First-order Markov Chain (FMC) as the baselines. Since we are particularly interested in scenarios where data is expensive to obtain and is insufficient (so as to evaluate sample efficiency), we did not compare with neural models for sequential modeling such as LSTM [Hochreiter and Schmidhuber, 1997], which generally require more data to perform well. We evaluated first with synthetic datasets randomly generated and then with real-world datasets. The same order of MMC, HMC, and MTD was used in the same evaluation setting. For each setting, we randomly split the data 75% for training and 25% for testing; we ran multiple trials for the average performance. All implementations are in Python. Since we observed that MMC with the greedy method performed equivalently to the exact method in many cases we tested, we chose to use this method for MMC results. Experiments were run on Paperspace C7 instances with 12 vCPUs and 30GB RAM. Implementation details and statistical test results are reported in the extended version [Zhang and Bucklew, 2022].

4.1 Synthetic Data

We compared MMC to the three baselines with three types of high-order data: 1) HMC data: data generated under the HMC assumption (Eq. (1)); 2) MMC data: data generated under the MMC assumption (Eq. (3)); 3) Causal data: data generated under the HMC assumption while assuming that the appearance of a state in any lag is associated with a high probability of generating the same state. For each type of data, we tested how the different models responded to changes in data size, state space size, and order size. Results for each evaluation setting are averaged over 30 trials. More details on how the data was generated and results for statistical tests are reported in the extended version [Zhang and Bucklew, 2022].

HMC Data

Since HMC data was generated under the HMC assumption (Eq. (1)), it is expected that MMC would not be able to handle HMC data well. The results are presented in Figs. 3, 4, and 5. First, the testing accuracy of all methods generally increased (albeit slightly) as the number of training data samples increased. We can see that HMC model performed poorly even after $20k$ data samples compared to others, illustrating its sample inefficiency. MTD dominated the others in almost all evaluation settings given that it explicitly considers influences from all lags, albeit in a rigid way. However, it used significantly more time than the other models for training. MTD appeared to be running linearly with respect to the data size but exponentially in the order and state sizes. MMC performed as badly as FMC (but both were more sample efficient than HMC), since both models make their specific assumptions about the data (which do not hold in HMC data). Since MTD was much more computationally expensive than the others, we further include two additional plots (Figs. 6 and 7) that exclude MTD to better compare the remaining methods. We can see the linear time performance of the remaining three methods w.r.t. the data size and that MMC is also linear in the state space size, which are consistent with our computational analysis.

MMC Data

The results with MMC data are presented in Figs. 8, 9, and 10. Results show that MMC indeed outperformed all the baselines under MMC data, which is one type of HMC data. What is more interesting was that MTD, again, used significantly more training time than the other models but only performed similarly to FMC: it mostly failed to account for the MMC data, which suggested a limitation of MTD models for modeling HMC data. HMC still suffered from sample inefficiency.

Causal Data

A common type of data that we may frequently encounter is causal data (such as our motivating example). Note that causal data only *approximately* matches with the MMC assumption. In particular, it does not prevent multiple lags to be

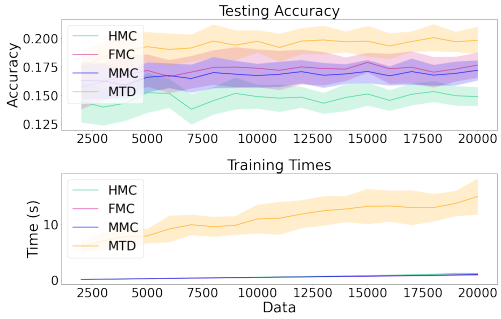

Figure 3: Results for HMC data while varying the data size. The state size is 7 and order size is 5.

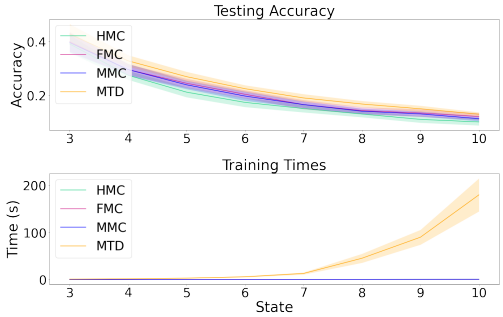

Figure 4: Results for HMC data while varying the state space size. The data size is 5k and order size is 5.

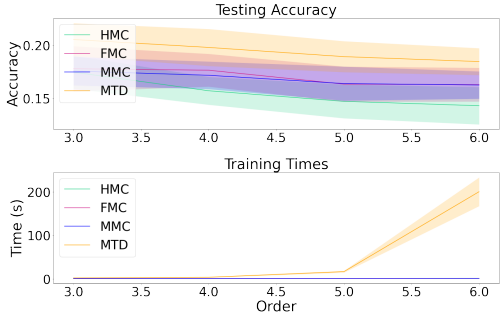

Figure 5: Results for HMC data while varying the order size. The data size is 5k and state size is 7.

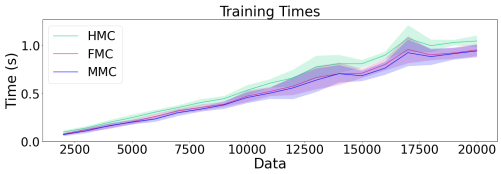

Figure 6: Time analysis for Fig. 3 without MTD.

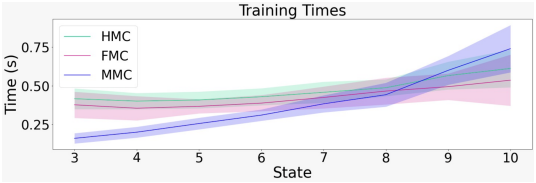

Figure 7: Time analysis for Fig. 4 without MTD.

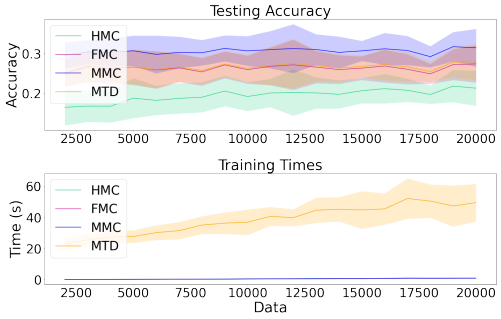

Figure 8: Results for MMC data while varying the data size. The state size is 7 and order size is 5.

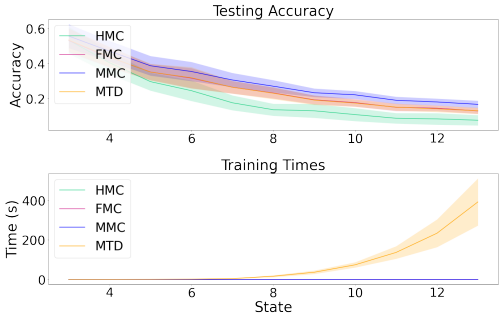

Figure 9: Results for MMC data while varying the state space size. The data size is $5k$ and order size is 5.

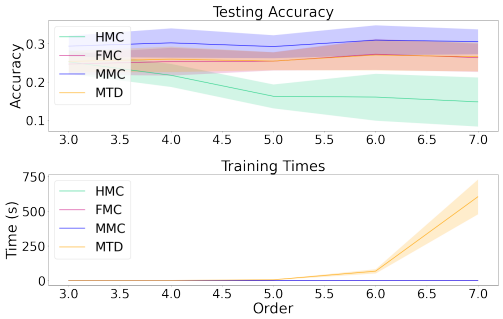

Figure 10: Results for MMC data while varying the order size. The data size is $5k$ and state size is 7.

correlated with the next state to be generated (i.e., denser correlations than MMC assumes). Hence, it imposes a challenge for MMC. The results with causal data are presented in Figs. 11, 12, and 13. Results show that MMC was able to generalize to this type of data quite well. In general, it outperformed the baselines on this type of data. In Fig. 11, you can see that MTD model caught up toward the end as more data was provided, which showed that MMC model was more sample efficient than MTD. This observation was further confirmed in Fig. 12, and 13, where MTD started being comparable to MMC but failed behind MMC as the state or order size increased (hence more training data would be needed).

4.2 Real-World Data

The real-world datasets were chosen since they are commonly used to evaluate sequence modeling with long-term dependencies. For example, the inflation dataset was used in MTD [Raftery, 1985].

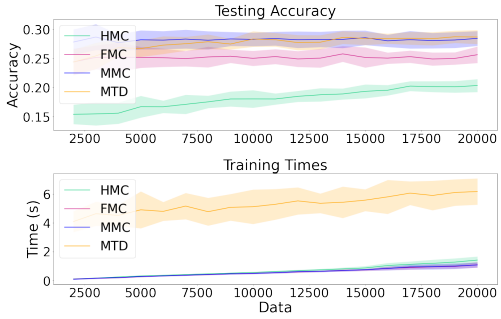

Figure 11: Results for causal data while varying the data size. The state size is 7 and order size is 5.

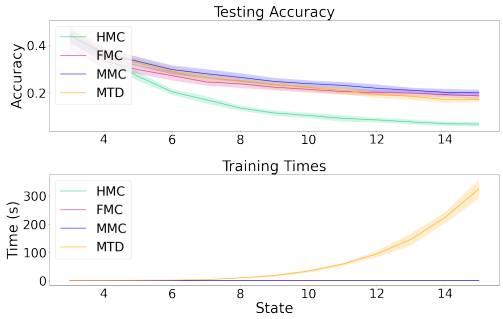

Figure 12: Results for causal data while varying the state space size. The data size is 5k and order size is 5.

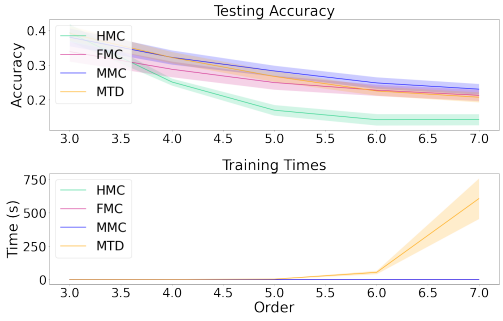

Figure 13: Results for causal data while varying the order size. The data size is 5k and state size is 7.

Inflation Data

In this experiment we use the inflation rates based on the US Consumer Price Index (CPI) from 1821 to 1999. This consists of a total of 179 years of inflation rates, one rate per year. In order to transform this data into a discrete state representation, we convert each year's rate based on how it compares to the standard deviation (STD) of inflation rates throughout the years. If the rate lies within 1 STD, we classify that year as either a small drop or rise. Similarly, between 1 STD and 2 STDs, 2 STDs and 3 STDs, and beyond 3 STDs, are classified as medium, large, and extra large drop or rise, respectively. It resulted in a total of 8 states. We chose order 3 for MMC, MTD, and HMC, resulting in 176 data samples. We ran 500 trials to compute the averages. Tab. 2 presents the results. We can see that MMC outperformed the three baselines. Analyzing the percentage of states generated by different lags, we can also see that the skipping links contributed to the better performance by substantially influencing state generation.

Model	Testing Accuracy
HMC	37.86%
FMC	47.84%
MMC	**50.76%**
MTD	48.15%

MMC-FMC	MMC-HMC	MMC-MTD
0.000	0.000	0.000

Lag 1	Lag 2	Lag 3
64.21%	22.75%	13.05%

Table 2: Results for inflation data based on US CPI (top left: accuracy; top right: p-values from student t-test; bottom: average percentages of states generated by each lag in testing data).

Model	Testing Accuracy
HMC	33.35%
FMC	40.13%
MMC	**43.01%**
MTD	40.94%

MMC-FMC	MMC-HMC	MMC-MTD
0.000	0.000	0.000

Lag 1	Lag 2	Lag 3	Lag 4	Lag 5
37.24%	23.10%	16.30%	12.75%	10.60%

Table 3: Results for Apple stock price data from NASDAQ (top left: accuracy; top right: p-values from student t-test; bottom: average percentages of states generated by each lag in testing data).

Apple Stock

We took it further to consider Apple stock price from NASDAQ using 1 hour interval. We obtained data consisting of a total of 3877 samples. We converted the data into a discrete state representation similar to that used in the inflation data. We chose order 5 for MMC, MTD, and HMC and ran 100 trials to compute the averages. Tab. 3 presents the results. Similar conclusions can be drawn here.

5 Conclusions and Discussions

In this paper, we introduced the Max Markov Chain (MMC) as a novel model for stochastic modeling. MMC was motivated by the limitations of the existing models and aimed at modeling domains with sparse correlations. We provided an analytical solution for parameter estimation. Based on this result, a greedy method was introduced. Comparison results with both synthetic and real-world datasets verified MMC as an alternative for stochastic modeling and highly scalable compared to competing models.

There exist plenty of opportunities for future research. Extensions of MMC to consider partial observability, multiple generation states, factored representations [Sallans and Hinton, 2004], and decision models [Russell, 2010], will be studied in future work. We will also investigate interesting applications of MMC, such as for intention recognition [Sukthankar *et al.*, 2014] and human behavior modeling [Zhang *et al.*, 2015].

Acknowledgements

This research is supported in part by the NSF grant 2047186 and the AFOSR grant FA9550-18-1-0067. The authors would also like to thank the anonymous reviewers for their helpful comments and suggestions.

References

[Baum and Petrie, 1966] Leonard E Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.

[Begleiter *et al.*, 2004] Ron Begleiter, Ran El-Yaniv, and Golan Yona. On prediction using variable order markov models. *Journal of Artificial Intelligence Research*, 22:385–421, 2004.

[Benjamin, 1979] Lorna Smith Benjamin. Use of structural analysis of social behavior (sasb) and markov chains to study dyadic interactions. *Journal of Abnormal Psychology*, 88(3):303, 1979.

[Berchtold and Raftery, 2002] André Berchtold and Adrian Raftery. The mixture transition distribution model for high-order markov chains and non-gaussian time series. *Statistical Science*, 17(3):328–356, 2002.

[Berchtold, 1995] André Berchtold. Autoregressive modelling of markov chains. In *Statistical Modelling*, pages 19–26. Springer, 1995.

[Bhattacharjya *et al.*, 2022] Debarun Bhattacharjya, Saurabh Sihag, Oktie Hassanzadeh, and Liza Bialik. Summary markov models for event sequences. *arXiv preprint arXiv:2205.03375*, 2022.

[Chin, 1977] Edwin H Chin. Modeling daily precipitation occurrence process with markov chain. *Water resources research*, 13(6):949–956, 1977.

[Cowles and Carlin, 1996] Mary Kathryn Cowles and Bradley P Carlin. Markov chain monte carlo convergence diagnostics: a comparative review. *Journal of the American Statistical Association*, 91(434):883–904, 1996.

[Elfeki and Dekking, 2001] Amro Elfeki and Michel Dekking. A markov chain model for subsurface characterization: theory and applications. *Mathematical geology*, 33(5):569–589, 2001.

[Fine *et al.*, 1998] Shai Fine, Yoram Singer, and Naftali Tishby. The hierarchical hidden markov model: Analysis and applications. *Machine learning*, 32(1):41–62, 1998.

[Galley, 2006] Michel Galley. A skip-chain conditional random field for ranking meeting utterances by importance. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 364–372, 2006.

[Gyftodimos and Flach, 2002] Elias Gyftodimos and Peter A Flach. Hierarchical bayesian networks: A probabilistic reasoning model for structured domains. In *Proceedings of the ICML-2002 Workshop on Development of Representations*, pages 23–30. Citeseer, 2002.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Jacobs and Lewis, 1978] Patricia A Jacobs and Peter AW Lewis. Discrete time series generated by mixtures. i: Correlational and runs properties. *Journal of the Royal Statistical Society: Series B (Methodological)*, 40(1):94–105, 1978.

[Kearns and Koller, 1999] Michael Kearns and Daphne Koller. Efficient reinforcement learning in factored mdps. In *IJCAI*, volume 16, pages 740–747, 1999.

[Metzner *et al.*, 2009] Philipp Metzner, Christof Schütte, and Eric Vanden-Eijnden. Transition path theory for markov jump processes. *Multiscale Modeling & Simulation*, 7(3):1192–1219, 2009.

[Milch *et al.*, 2005] Brian Milch, Bhaskara Marthi, David Sontag, Stuart Russell, Daniel L Ong, and Andrey Kolobov. Approximate inference for infinite contingent bayesian networks. In *International Workshop on Artificial Intelligence and Statistics*, pages 238–245. PMLR, 2005.

[Parzen, 1999] Emanuel Parzen. *Stochastic processes*. SIAM, 1999.

[Pearl, 2003] Judea Pearl. Statistics and causal inference: A review. *Test*, 12(2):281–345, 2003.

[Pentland and Liu, 1999] Alex Pentland and Andrew Liu. Modeling and prediction of human behavior. *Neural computation*, 11(1):229–242, 1999.

[Petropoulos *et al.*, 2017] Anastasios Petropoulos, Sotirios P Chatzis, and Stelios Xanthopoulos. A hidden markov model with dependence jumps for predictive modeling of multidimensional time-series. *Information Sciences*, 412:50–66, 2017.

[Raftery and Tavaré, 1994] Adrian Raftery and Simon Tavaré. Estimation and modelling repeated patterns in high order markov chains with the mixture transition distribution model. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 43(1):179–199, 1994.

[Raftery, 1985] Adrian E Raftery. A model for high-order markov chains. *Journal of the Royal Statistical Society: Series B (Methodological)*, 47(3):528–539, 1985.

[Randall, 2006] Dana Randall. Rapidly mixing markov chains with applications in computer science and physics. *Computing in Science & Engineering*, 8(2):30–41, 2006.

[Roucos *et al.*, 1982] S Roucos, J Makhoul, and R Schwartz. A variable-order markov chain for coding of speech spectra. In *ICASSP'82. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 7, pages 582–585. IEEE, 1982.

[Russell, 2010] Stuart J Russell. *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010.

[Sallans and Hinton, 2004] Brian Sallans and Geoffrey E Hinton. Reinforcement learning with factored states and

actions. *The Journal of Machine Learning Research*, 5:1063–1088, 2004.

[Schmidt, 2000] Bernard Schmidt. *The modelling of human behaviour*, volume 132. Society for Computer Simulation International, 2000.

[Stewart, 1994] William J Stewart. *Introduction to the numerical solution of Markov chains*. Princeton University Press, 1994.

[Sukthankar *et al.*, 2014] Gita Sukthankar, Christopher Geib, Hung Bui, David Pynadath, and Robert P Goldman. *Plan, activity, and intent recognition: Theory and practice*. Newnes, 2014.

[Sutton and McCallum, 2004] Charles Sutton and Andrew McCallum. Collective segmentation and labeling of distant entities in information extraction. In *ICML Workshop on Statistical Relational Learning and Its Connections to Other Fields*, 2004.

[Sutton *et al.*, 1999] Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

[Ye *et al.*, 2004] Nong Ye, Yebin Zhang, and Connie M Borror. Robustness of the markov-chain model for cyber-attack detection. *IEEE transactions on reliability*, 53(1):116–123, 2004.

[Zhang and Bucklew, 2022] Yu Zhang and Mitchell Bucklew. Max markov chain. *arXiv preprint arXiv:2211.01496*, 2022.

[Zhang *et al.*, 2015] Yu Zhang, Sarath Sreedharan, and Subbarao Kambhampati. Capability models and their applications in planning. In *AAMAS*, pages 1151–1159, 2015.