AUTHOR'S VERSION. © 2025 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses.

Theory Meets Practice: Assessment of Decomposition-Based Approximation in 2D Markov Models for Server Cluster Resilience

Carina Baur, Frank Loh, Tobias Hoßfeld
University of Würzburg, Institute of Computer Science, Würzburg, Germany
Contact: carina.baur@uni-wuerzburg.de

***Abstract*—Markov models are widely used to analyze complex stochastic systems, particularly in modeling of server cluster resilience, where they provide valuable insights into key performance metrics such as unavailability and blocking probability. However, their applicability to large-scale systems is often constrained by the state space explosion problem, leading researchers to employ decomposition-based approximation techniques to improve computational efficiency. Some commonly used approaches involve nearly-completely decomposable (NCD) Markov models, where processes operating at significantly different time scales are treated separately.**

This work critically examines the assumption that a large difference in time scales between system components guarantees a small approximation error. Using two examples from the context of server cluster resilience, we demonstrate that simplifications in NCD Markov models can lead to substantial inaccuracies. While decomposition-based methods offer large computational savings, their accuracy depends on specific structural conditions, rather than just the separation of time scales. Our findings emphasize the importance of carefully validating approximation assumptions and model structures to ensure reliable performance evaluation in large-scale stochastic systems.

***Index Terms*—Markov model, decomposition, nearly-completely decomposable, approximation, resilience modeling**

I. Introduction

Modern networks are increasingly complex and large-scale, yet they must remain resilient and be able to recover quickly after experiencing faults, to handle current and future workloads efficiently. Ensuring their resilience requires a deep understanding of system behavior, which is often achieved through mathematical modeling. Markov models serve as a fundamental tool in this regard, offering a structured way to abstract real-world complexity while capturing essential system dynamics. Despite their effectiveness, Markov models can become computationally infeasible due to the state space explosion problem. To address this, researchers frequently rely on simplifying assumptions and approximation techniques that reduce computational costs and improve runtime performance. Some approaches involve nearly-completely decomposable (NCD) systems, which leverage the separation of time scales between different system processes [1]. For example, in network models, customer arrival and service typically occur at much higher rates than server failures and repairs. Intuitively, one might assume that when analyzing customer behavior, the slower server dynamics can be evaluated separately, reducing the model to two one-dimensional approximated models. Another argument is that if server faults are relatively seldom, the system is in a fully functioning state for most of the time, tempting researchers to only investigate this state.

This type of assumption is not uncommon in system modeling. However, while decomposition-based methods are well-established and theoretically justified under specific conditions, the necessary assumptions for accurate approximations are sometimes overlooked. Researchers have derived formal bounds on approximation errors [2], but intuitive reasoning often leads to the incorrect assumption that a large separation in time scales guarantees a small approximation error.

In this work, we challenge this assumption by demonstrating that it can lead to substantial errors in practice. Using two examples from the context of server cluster resilience, we show how simplifications in NCD Markov models can result in significant inaccuracies. Consequently, our contribution is threefold: (1) We present a simple approximation method similar to known aggregation based approximations, (2) we use two illustrative examples to investigate how incorrect assumptions in NCD Markov models can lead to significant inaccuracies, (3) and we provide our implementation for reproducibility and further usage [3]. With these contributions, we answer the following research question in this work: *Does a significantly high difference in the time-scale of two processes guarantee a small approximation error when decomposing the model?* By investigating this question, we provide insights into the risks of over-simplification in Markov model approximations, emphasizing the importance of carefully validating assumptions in theoretical and applied settings.

The remainder is structured as follows: Section II introduces related work, Section III background, and Section IV the approximation method. Section V defines two scenarios, related to real-life applications of server cluster resilience. Section VI evaluates, demonstrates the influence of system parameters and structure on the approximation error, and analyzes the runtime improvement as the main reason for using such an approximation. Finally, Section VII presents a discussion for practical implications and concludes.

Paper is accepted for publication at RNDM 2025, Trondheim, Norway – the 15th International Workshop on Resilient Networks Design and Modeling.

AUTHOR'S VERSION. © 2025 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses.

II. RELATED WORK

Markov models in general, and two-dimensional Markov models in particular, have been extensively studied in the literature. Tran-Gia and Hoßfeld [4] provide a general understanding of queuing theory and Markov models, illustrating their application in various contexts, including software-defined networking and Internet of Things (IoT) applications. A foundational theoretical work is given by Kemeny and Snell [5]. Metzger et al. [6] offer guidelines for effectively leveraging the Markov property of arrival processes in large IoT deployments. Additional applications are given by Loh et al. [7], who use a Markov model to assess the Quality of Experience for video streaming. Brown et al. [8] introduce Hierarchical Markov Modeling (HMM) as a method for distribution system reliability assessment, marking a significant evolution from simpler reliability models. Buchholz [9] tackles hierarchical Markov models from a computational perspective, addressing challenges posed by the large state spaces of Markov chains and proposing reduction methods based on symmetry and structural properties. Muscariello et al. [10] survey Internet traffic modeling and introduce a hierarchical approach that utilizes a two-dimensional Markov model and rate matrix structures.

A more refined class of Markov models known as quasi birth-death processes (QBDPs) has been widely investigated. Bright et al. [11] extend classical QBDPs to Level-Dependent Quasi Birth-Death Processes (LDQBDPs) and present algorithms for computing their steady states. Further approximations for Markov models in general and LDQBDPs in particular are explored in [12], which introduces a truncation-based approximation technique. An efficient algorithm for analyzing both finite and infinite models is presented in [13]. Baumann et al. [14] extend LDQBDPs with a matrix-analytical approach for LDQBDPs with so-called catastrophes.

Decomposition strategies have been employed to approximate Markov models, addressing computational challenges associated with large state spaces. Heegaard et al. [15] utilize time-space decomposition to analyze network survivability. Somani et al. [16] introduce state merging and independent sub-model analysis techniques. Woodward [17] proposes a decomposition method that separates a Markov chain into two independent sub-chains, reducing computational complexity.

Theoretical works focus on decomposability and lumpability in Markov models, with an emphasis on NCD systems. An extensive overview and a thorough theoretical analysis of approximation error bounds and prerequisites is given by Courtois [1]. Simon and Ando [18] introduce an aggregation framework for dynamic systems with nearly-decomposable structures, showing how short-run and long-run dynamics influence system behavior. The concept of lumpability, a criterion for the exact aggregation of microstates into macrostates, is investigated in multiple works [19], [20], [21], [22].

In contrast to related work, this study examines the potential inaccuracies of aggregation-based approximation methods when not applied carefully. We highlight possible pitfalls using an intuitive approximation approach that may be favored by researchers due to its simplicity, computational efficiency, and apparent alignment with the aforementioned theoretical frameworks.

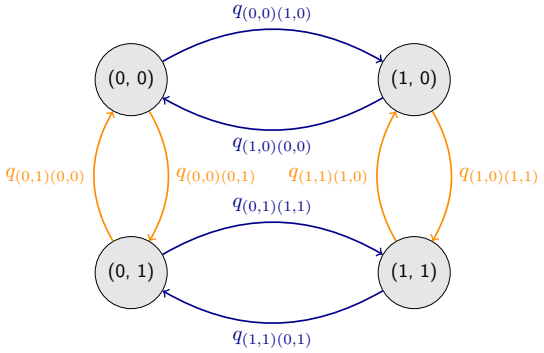

Figure 1: Example for a two-dimensional Markov model.

III. BACKGROUND

This work focuses on two-dimensional continuous-time Markov chains (CTMCs), which are a stochastic process with the Markov property. A stochastic process has the Markov property, when its future development only depends on the current state. Figure 1 illustrates the structure of these models using a state-transition diagram. They are defined on a continuous time space T and a finite, discrete, and two-dimensional state space S. Each state s in this state space is denoted by its X- and Y-coordinate with $s = (x, y) \in S$ or by an index $i \in \{1, \ldots, n\}$, with $n = |S|$. In the approximation approach, this two-dimensional Markov model is decomposed into Y_m layers with $y \in \{0, \ldots, Y_m - 1\}$. Each layer consists of X_m states, i.e., $x \in \{0, \ldots, X_m - 1\}$. Thus, the two-dimensional Markov model consists of a total of $n = Y_m \cdot X_m$ states.

Rate Matrix: The transition rate between two states $i = (x_1, y_1)$ and $j = (x_2, y_2)$ is denoted by $q_{(x_1,y_1)(x_2,y_2)} = q_{ij}$. The transitions are described by a rate matrix $Q \in \mathbb{R}^{n \times n}$, with the rates $q_{ij} \geq 0$ for $i \neq j; i, j \in S$. For $i = j$, the entry on the diagonal is defined as $q_{ii} := -\sum_{j \in S, j \neq i} q_{ij}$. Therefore, the row-wise sum of the rate matrix Q is always zero.

State Equation System: To derive meaningful statements from this theoretical model about the underlying system, the steady-state probabilities of the model are derived. The steady-state probability of each state i is denoted with $w(i)$. Since, in this paper, we are interested in investigating systems that are already in a steady state, the state probabilities are neither dependent on time nor on an initial state. Thus, the state probabilities have reached a statistical equilibrium. The state probabilities are accumulated in a single vector

$$W = \begin{pmatrix} w(1) & w(2) & \ldots & w(n) \end{pmatrix}. \tag{1}$$

Using the statistical equilibrium, which means incoming and outgoing flows for each state need to be equal so that state

AUTHOR'S VERSION. © 2025 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses.

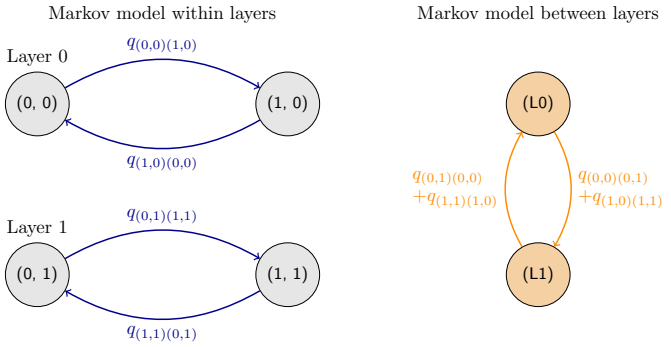

Figure 2: Illustration of a decomposition for the example in Figure 1 into a Markov model per layer (Eq. (4)) and a Markov model between layers (Eq. (6)).

probabilities stop changing, we can calculate this vector by solving the state equation system represented by

$$WQ = 0, \tag{2}$$

with the normalization condition

$$\|W\|_1 = 1. \tag{3}$$

We indicate the 1-norm of a vector $W \in \mathbb{R}^n$ with $\|W\|_1 := \sum_{i=1}^{n} |w_i| = \sum_{i=1}^{n} w_i$, since $w_i \geq 0$.

IV. Divide and Conquer: Decomposition Method

This section details the approximation approach and how the steady state probabilities are computed. A visualization is shown in Figure 2 and the implementation of the approach is available in [3].

A. Divide

Let S be the state space of a Markov model. A layer L_i is a set of states in a Markov model that share the same Y-coordinate, formally $L_i = \{(x, y) \in S | y = i\}$. Let $L = \{L_i\}$ be the set of all layers. Note that in the regarded model structures, each node is part of a layer and each layer has the same size. We assume only transitions between neighboring layers, A single layer is considered as its own Markov model by ignoring all transitions that come from a state outside the layer or leave that layer. With this definition, the rate matrix Q of the complete model is structured as

$$Q = \begin{pmatrix} Q_0 & Q_{01} & 0 & \dots & 0 & 0 \\ Q_{10} & Q_1 & Q_{12} & \dots & 0 & 0 \\ 0 & Q_{21} & Q_2 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & Q_{l-1} & Q_{l-1,l} \\ 0 & 0 & 0 & \dots & Q_{l,l-1} & Q_l \end{pmatrix},$$

with $Q_i \in \mathbb{R}^{|L_i| \times |L_i|}$ as the rate matrix of the layer L_i. For Q_i to be a valid rate matrix with row sums of zero, the diagonal elements of the original rate matrix Q must be adjusted accordingly for the layer models. The elements $Q_{ij} \in \mathbb{R}^{|L_i| \times |L_j|}$ are matrices containing the rates for the transitions from layer L_i to layer L_j. These matrices do not have a row wise sum of zero and are not rate matrices.

We begin by solving each layer independently using the state equation system with the adjusted rate matrices Q_i. This results in an approximate solution for the conditional probability to be in the state (x, i) under the assumption that the system is already within the correct layer. We denote this probability with $w_i(x)$ and accumulate the probabilities for each layer in a vector W_i.

Definition IV.1 (State equations per layer)**.** The state equation system of each layer L_i in the steady-state is given by

$$W_i Q_i = 0, \tag{4}$$
$$\|W_i\|_1 = 1. \tag{5}$$

The second step of the approach is to approximate the probability $l(y)$ of being in a certain layer y. We propose to collapse each layer into a single state and interpret them as a new, one-dimensional Markov model which can be solved efficiently. The transitions between those new states are modeled by summing up all transition rates between states of the respective layers. The sum is chosen, as the ratio between the rates leaving and entering the state is most important to capture as accurately as possible. Multiplying all rates by a constant factor does not affect state probabilities, as the factor cancels out in the rate matrix and the solution of the state equation. In reality, this step only results in a Markov model when the complete system is lumpable [19]. Then, the layer probability calculated here is accurate.

Let $A \in \mathbb{R}^{m \times n}$ be a matrix. We define $s(A) := \sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij}$ as the operation that results in the sum of all elements of a matrix. Then, the rate matrix for this collapsed system is

$$Q_l = \begin{pmatrix} -q_0 & s(Q_{01}) & 0 & \dots & 0 & 0 \\ s(Q_{10}) & -q_1 & s(Q_{12}) & \dots & 0 & 0 \\ 0 & s(Q_{21}) & -q_2 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & -q_{l-1} & s(Q_{l-1,l}) \\ 0 & 0 & 0 & \dots & s(Q_{l,l-1}) & -q_l \end{pmatrix},$$

where q_i is the sum of the entries in the respective row, to ensure each row sum is equal to zero. It holds that $Q_l \in \mathbb{R}^{l \times l}$, with l as the number of layers. Then, to generate the rate matrix for the layer system, a simple addition of the respective matrix elements of the original rate matrix is required. We calculate the vector L with the accumulated layer probabilities $l(i)$ by solving the state equation system for this collapsed model.

Definition IV.2. The state equation system for the collapsed Markov model is given by

$$LQ_l = 0, \tag{6}$$
$$\|L\|_1 = 1. \tag{7}$$

AUTHOR'S VERSION. © 2025 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses.

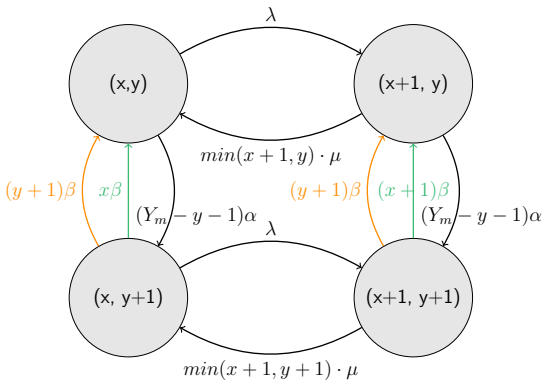

Figure 3: Structure of Sc. RL1 (orange) and Sc. RL2 (green).

B. Conquer

This section explains how the two previous simplifications are integrated into a complete solution and defines the target metrics. For each state $s = (x, y) \in S$, the true state probability $w(s)$ is approximated with the probability denoted as $\overline{w}(s)$. They are calculated by $\overline{w}(s) = l(y) \cdot w_y(x)$ and can be accumulated in a vector $\overline{W}$. Our goal is then the evaluation of the approximation error, given by $E(\overline{W}, W)$, with $E(x, y)$ as an error metric comparing two vectors of same size. We evaluate multiple error metrics. First, we choose the maximum absolute error (MAE), calculated with $\max(|y_i - x_i|)$ for two vectors $x, y \in \mathbb{R}^n$, which means that the error for the complete model is the maximum of the absolute errors for each state probability. Consequently, we achieve a worst case evaluation for our approximation error. Additionally, we assess three use case-specific metrics: the unavailability probability, the blocking probability, and the mean power draw. The unavailability probability represents the likelihood that all servers in the system are down and the service is no longer available. This corresponds to the probability of layer 0. The blocking probability quantifies the likelihood of the system being at full capacity, which means that a new customer is blocked. It is defined as the sum of the probabilities of all final states in each layer. The mean power draw is evaluated with a simplified power model, which is introduced in detail in Section VI-C. We calculate all three metrics for the real and approximated model and evaluate their absolute difference.

V. Real-Life Scenarios: Resilience of Server Clusters

This section defines two scenarios related to real-life applications and evaluates them with respect to the approximation accuracy. These scenarios are chosen from the context of resilience modeling. Therefore, we focus on modeling server faults and system behavior after a fault.

A. Scenario Definition

To demonstrate how, at first glance, small differences in the model can significantly impact the approximation error, we investigate two models corresponding to real example server systems.

The model structure is visualized in Figure 3, with the common characteristics in black and the defining changes between the two scenarios in orange and green, respectively. In both scenarios the arriving customers are modeled with a Poisson process with arrival rate λ in the X-direction of the model. The layer determines the available resources, meaning that for y available servers and x customers in the system, the transition rate to state $(x-1, y)$ due to a completed service is given by $min(x, y) \cdot \mu$. The process in the Y-direction describes the changing of the availability of servers. We assume that in case of a fault, each faulty server is restored with a rate of α. This means for y available servers, the $Y_m - y - 1$ servers under repair have a total restoration rate of $(Y_m - y - 1)\alpha$.

In Scenario RL1, called "independent server failure", each server experiences a system-fault with a rate of β, removing this server from the available servers. This is modeled with a Y-dependency of β-transitions, as we assume the faults to be independent of whether the server is currently being used by a customer. For y active servers the total failure rate is $y\beta$.

In Scenario RL2, called "load-dependent server failure", we extend Scenario RL1 and assume that increased load on a server increases its fault rate. This reflects real-life conditions, where increased heating accelerates hardware wear and tear, making servers more susceptible to failures [23]. In our simplified model, this implies that if the system is currently in state (x, y), the fault rate—defined as the transition rate from state (x, y) to state $(x, y-1)$—is given by $y \cdot \frac{x}{y} \cdot \beta = x \cdot \beta$. This means that the default fault rate is increased by a factor of the load per server, which results in an X-dependent β. All other transitions remain the same as in Scenario RL1.

The variable parameters for these scenarios are the system constants λ, μ, α, and β, corresponding to the arrival and service rate of customers, and the restoration and fault rate of servers. The ratios between these constants are more important than their value, as constant factors multiplied with all four values cancel out. To investigate realistic scenarios while truncating the parameter range to a manageable size we choose to fix the customer service rate μ to $100,000$ and the server restoration rate α to 100. To put these values into perspective: Assuming a mean customer service time of $1/\mu = 1\,\text{s}$, a server is typically restored within 15 minutes after a fault. For the customer arrival rate we choose values between $10,000$ and $100,000$ with a step size of $10,000$. This enables us to investigate different ratios of λ and μ, corresponding to the offered load in one-dimensional Markov models, between 0.1 and 1 and results in a mean customer inter arrival time between 1 s and 10 s. For the server fault rate β, we stretch values beyond realistic values to demonstrate the influences of the order of magnitude between the two processes. We choose a logarithmic scaling and values between $100,000$ and 1, resulting in a mean time to failure for a single server between 1 s and almost 28 h. Since μ is set to $100,000$, smaller β values are more realistic and lead to a larger order of magnitude between the two processes.

We set the maximum number of servers to five and the maximum number of customers to 500. This results in an

AUTHOR'S VERSION. © 2025 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses.

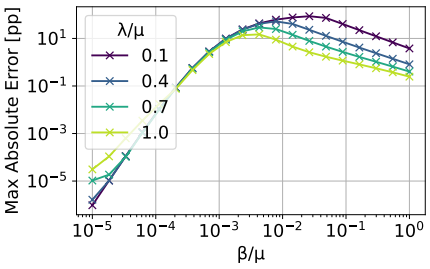

Figure 4: Influence of β and λ, normalized with μ, on the MAE in RL1.

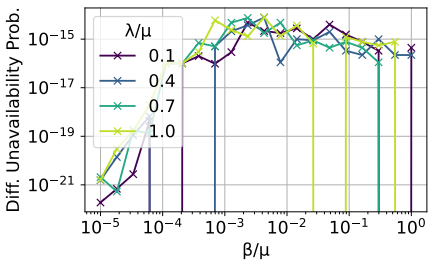

Figure 5: Influence of β and λ, normalized with μ, on the unavail. prob. in RL1.

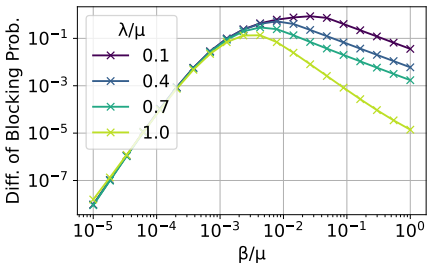

Figure 6: Influence of β and λ, normalized with μ, on the block. prob. in RL1.

overall system size of 2500 states.

VI. Evaluation

In this section we evaluate both scenarios and demonstrate the high dependency of the error on the model structure and the investigated parameters. We will see that the error can be large, even if the two processes in the model are a significant order of magnitude apart. Additionally, we introduce a simple power draw model to assess the approximation error on power metrics and conduct a runtime comparison to investigate why such an approximation is appealing to researchers.

A. Scenario RL1

Figure 4 shows the MAE in percentage points (pp) on the y-axis, the parameter β on the x-axis, and the parameter λ color-coded. Both parameters are normalized with the value of μ to show the order of magnitude differences. We observe that the MAE is highly dependent on the β value. The approximation achieves negligible small errors for $\beta/\mu = 10^{-5}$ and $\lambda/\mu = 0.1$, but reaches large errors of up to 86 pp for $\beta/\mu = 2.6 \cdot 10^{-2}$ and $\lambda/\mu = 0.1$. This is a surprising result, as such a difference is typically considered to be large enough to decouple both processes. Further increasing β reduces the error again. The parameter λ also shows an influence on the MAE. This influence is also interacting with the value of β, since for small β values, a smaller λ value leads to smaller errors, but for larger β values, the larger λ values are favorable.

The evaluation above demonstrates the worst case by investigating the MAE. When applying Markov models to analyze server cluster resilience, typically only sets of states with certain meaning are of primary interest. Therefore, we evaluate the unavailability probability, i.e., the probability of being in layer zero, in Figure 5. The figure shows the absolute difference of the unavailability probability between the approximation and the real model on the y-axis, the parameter β on the x-axis, and the parameter λ color-coded. Again, both parameters are normalized with the value of μ. We see that negligible small numerical errors are achieved in this case for all values of λ and β. This is not the case for all metrics that sum up multiple state probabilities, as we can see in Figure 6. The figure shows the absolute difference of the blocking probability between the approximation and the real model on the y-axis. Like in Figure 4 we observe a strong influence of β on the error, and a smaller influence of λ, only for larger β values. The error of the blocking probability approximation peaks for $\beta/\mu = 2.6 \cdot 10^{-2}$ and $\lambda/\mu = 0.1$, reaching a large absolute difference between the probabilities of 0.86.

We can conclude for this Scenario, that the error is highly dependent on the values of β, and a small dependency is observed for λ. Furthermore, against general intuition, a higher order of magnitude between both processes does not guarantee a small overall error or a small error for the blocking probability, as we achieve significantly high errors for a difference of 10^{-2}. On the other hand, the unavailability probability can be approximated accurately in this scenario.

B. Scenario RL2

We now evaluate Scenario RL2, which differs from Scenario RL1 in the dependency of the β-transitions only. Figure 7 displays the MAE on the y-axis, with the parameter β plotted along the x-axis and λ represented using color coding. Both β and λ are normalized by μ to highlight differences in their orders of magnitude. Just as in Scenario RL1, we observe a strong correlation between the error and β, and a smaller influence of λ. The highest error of 92 pp is achieved for the smallest β/μ ratios, which means, that even with an order of magnitude of 10^{-5}, the approximation is not usable. Against expectation, larger values of β lead to smaller errors, with the lowest error of $8.0 \cdot 10^{-6}$ achieved for $\beta/\mu = \lambda/\mu = 1$.

Figure 8 presents the absolute difference of the unavailability probability for Scenario RL2, which was accurately approximated in Scenario RL1. The influence of β follows a similar trend as observed for the MAE, just with an initial increase in error followed by a decrease as β increases. The maximum absolute difference of 0.92 occurs for $\beta/\mu = 2 \cdot 10^{-4}$ and $\lambda/\mu = 0.1$. The parameter λ has only a minor impact on the results. A similar pattern is observed in Figure 9 for the absolute difference of the blocking probability, meaning that a high order of magnitude between the processes leads to large errors for both metrics.

Our evaluation of both scenarios demonstrates that the approximation error is highly dependent on the system structure and the investigated parameters, and that large errors can occur, even if the order of magnitude between the two processes is high.

Consequently, we can answer our research question based on our real-life scenario analysis as follows. *A high order of*

AUTHOR'S VERSION. © 2025 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses.

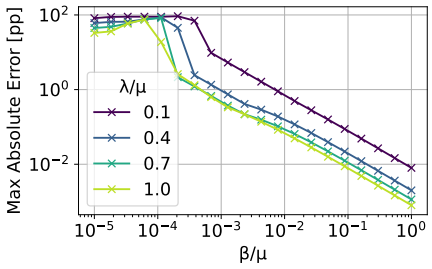

Figure 7: Influence of β and λ, normalized with μ, on the MAE in RL2.

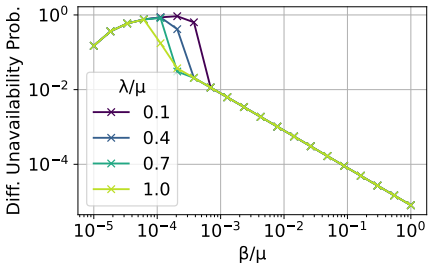

Figure 8: Influence of β and λ, normalized with μ, on the unavail. prob. in RL2.

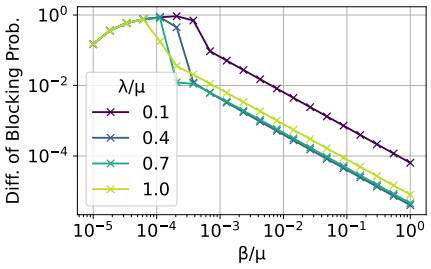

Figure 9: Influence of β and λ, normalized with μ, on the block. prob. in RL2.

magnitude between the two processes in a two-dimensional Markov model does not guarantee a small approximation error in a divide-and-conquer based approach. The system structure, as well as the parameters have a strong influence on the error. Additionally, parameter interaction is observed.

C. Power Draw Modeling

While the previous analysis focused on the blocking and unavailability probability—two metrics that primarily capture aspects of resilience and service quality—it is equally important to consider energy efficiency, especially in systems like server clusters where energy and power represent a significant operational cost factor and design constraint. Therefore, in this section, we introduce a simple power draw model to evaluate the mean power draw with regards to its approximation error.

Assessing power draw is a necessary step toward understanding overall energy consumption, as it reflects the system's instantaneous energy use, which accumulates over time into total energy costs. This is particularly relevant in energy-aware system design, where minimizing consumption without sacrificing performance is a key goal. Moreover, including power draw models lays the groundwork for studying trade-offs between resilience, service quality, and energy efficiency. These trade-offs are critical in many real-world applications, where improving one aspect often comes at the expense of another, and informed design decisions must balance all three dimensions. We therefore assess the influence of the simplifications made by the approximation on the power draw metric.

To assess the mean power draw of the system, we adopt a simplified power model that assigns fixed power values to servers based on their operational state. Our example systems consist of five servers, as defined in Section V-A, each of which we assume to be in one of three modes: active, idle, or non-operational. We assume that an active server draws 100 units of power, an idle server draws 50 units, and a non-operational server draws 0. Since we focus on the approximation error and not the results for the power draw, these values were chosen to only resemble realistic values, with an idle power of 50 % of the maximum power draw. These values can be replaced with those from a real system if needed. For a given system state (x, y), the number of active servers is computed as $\min(x, y)$, while the number of idle servers is given by $y - \min(x, y)$. The power draw in each state is thus calculated as $100 \cdot \min(x, y) + 50 \cdot (y - \min(x, y))$. The overall mean power draw is then obtained by weighting the power draw in each state by its corresponding stationary probability and summing across all possible states.

The results for this power evaluation can be seen in Figure 10 and Figure 11. Figure 10 shows the absolute difference in the calculated mean power draw between the exact model and the approximation for Scenario RL1 across varying values of β/μ on the x-axis, and for different load intensities λ/μ color coded. The error varies significantly with the failure rate β, and peak errors of over 10 power units occur around $\beta/\mu = 10^{-2}$. It can also be seen, that the error is negligible small for both high and low values of β. Figure 11 shows the absolute difference in the calculated mean power draw between the exact model and the approximation for the RL2 system, plotted against β/μ on the x-axis and for various values of λ/μ color coded. In contrast to RL1, the approximation error in RL2 is substantial for low failure rates ($\beta/\mu < 10^{-3}$), with the error exceeding 200 units in some cases. As β/μ increases, the error rapidly drops and becomes negligible for $\beta/\mu > 10^{-3}$.

Our previous analysis demonstrated that a large order of magnitude between the two stochastic processes does not necessarily guarantee a small approximation error. In fact, substantial errors are observed even when the rates differed by several orders of magnitude. The power draw evaluation further reinforces this finding, as it reveals significant deviations in mean power consumption under certain parameter configurations, particularly in regions where one might expect the approximation to perform well due to apparent time-scale separation, as for small β values in Scenario RL2.

D. Computation Time Comparison

In this section, we theoretically calculate the expected runtime and practically measure runtime improvement of our approximation approach, showcasing why such an approximation is appealing to researchers. Additional benefits, like possible parallelization and a simplified modeling process, are not included in this evaluation. However, because the layers of our models can be computed independently using the proposed approach, we expect further improvement potential utilizing parallelization.

In the theoretical analysis we compare the computational effort for solving the state equation system, for now ignoring

AUTHOR'S VERSION. © 2025 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses.

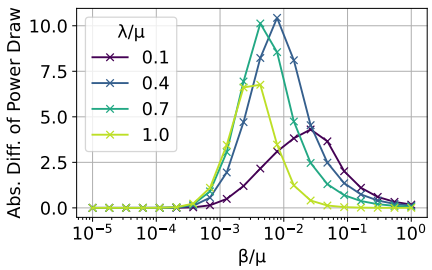

Figure 10: Absolute difference of the calculated power draw in RL1.

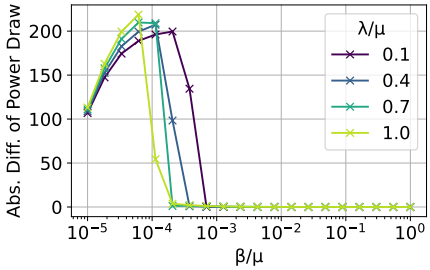

Figure 11: Absolute difference of the calculated power draw in RL2.

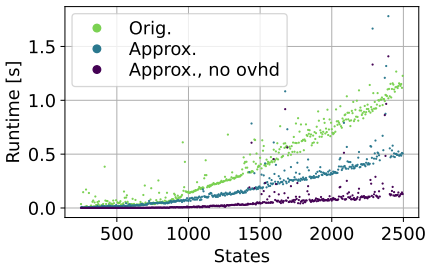

Figure 12: Runtime: original and approx. with and without overhead in Sc. RL1.

additional overhead. We assume that the matrix inversion has a complexity of $O(n^3)$. As this affects both the original calculation and the approximation, we ignore possible improvements for certain matrix structures. The main benefit of the approximation are the smaller matrix sizes. Let $x \in \mathbb{N}$ be the number of states per layer and let $y \in \mathbb{N}$ be the number of layers in the system. Let $n := x \cdot y$ be the number of states. For the original solution, one matrix with dimension $n \times n$ is inverted. This part of the calculation is estimated with costs of $C_{orig} := n^3 = (x \cdot y)^3$. For the approximation, this is divided into a matrix with dimension $y \times y$ and y matrices with dimension $x \times x$, which are inverted. Overall, this results in estimated costs of $C_{approx} := y^3 + y \cdot x^3$. With

$$\begin{aligned} & C_{orig} = (x \cdot y)^3 \geq y^3 + y \cdot x^3 = C_{approx} \\ \Leftrightarrow \quad & x^3 \cdot y^2 \geq x^3 + y^2 \\ \Leftrightarrow \quad & (x^3 - 1)(y^2 - 1) \geq 1, \end{aligned}$$

we conclude that the approximation is faster than the original calculation for all models larger than a 1×1 system. Investigating the theoretically saved runtime with

$$\begin{aligned} \frac{C_{approx}}{C_{orig}} &= \frac{2 \cdot (y^3 + y \cdot x^3)}{2 \cdot (x \cdot y)^3} \\ &= 1 - \frac{1}{x^3} - \frac{1}{y^2}, \end{aligned}$$

we achieve 62.5 % time savings for the computation considering a 2×2 system and 98.9 % for a 10×10 system. In addition, the influence of x, and therefore the influence of the number of states per layer, is greater than the influence of the number of layers y.

To further investigate this result, we conduct a runtime measurement, focusing on the process time for three different parts of the implementation. First, we measure the time to solve the original complete model. Second, we measure the time to calculate the approximation with all overhead, like initializing additional arrays, dividing the model, and recombining the solutions. Third, we measure only the computation time required to solve the smaller state equation systems in the approximation. While measurements are performed for both scenarios, we focus on Scenario RL1 for a detailed evaluation, since the results for Scenario RL2 do not differ significantly. For our example assessment we use Python 3.12.3 on a machine with Ubuntu 24.04, an Intel Core i7-7700, and 32 GB RAM. For reproducibility and further usage, the code for our assessment is available in [3]. The solution of the system is implemented using linalg.solve, since we are interested in a comparison and not in the most efficient implementation. The results for Scenario RL1 can be seen in Figure 12. The figure shows the size as the number of states on the x-axis and the runtime in seconds on the y-axis. The green values show the measurements for the original solution, the blue values for the approximation with overhead, and the purple values for the approximation without any overhead.

The green values for the original calculation start with a minimum value for a system size of 250 states with a runtime of 0.002 s and reach 1.227 s for the largest system size, with few outliers. In general, we see an exponential increase of the runtime. The blue data points show the values for the approximation with overhead, which start at a minimum value of 0.004 s and increase exponentially to a maximum of 0.506 s, not considering outliers. While we still see an exponential increase, the original curve is steeper and, therefore, our approximation is more efficient. For our largest system, containing 2500 states, the approximation is 60 % faster than the original solution and further speedup is expected for larger models. Without overhead, the purple values start with a minimum of 0.001 s and reach a maximum of 0.129 s. The curve increases exponentially. This results in a total speedup of 90 % for the matrix calculations. In our theoretical analysis, we calculated the improvement of the approximation without any overhead in contrast to the original calculation with almost 99 %. This difference is explained with technical limitations in the implementation, the measurement setup, and additional necessary operations that are not considered above, like transposing matrices.

VII. Conclusion and Discussion

We demonstrate that while decomposition-based approximations provide significant computational advantages, they can introduce substantial errors if applied without careful validation. The errors reached from negligible small numerical errors of magnitude 10^{-15} to significant errors of 92 pp and are highly dependent on the system structure and the investigated parameters and metrics. Many similar methods already exist, each with well-defined conditions for ensuring accurate results. However, our study highlights that the intuitive assumption

AUTHOR'S VERSION. © 2025 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses.

that a large difference in time scales between processes guarantees a small approximation error is incorrect in a general case.

A useful analogy for understanding this issue is given by Simon and Ando [18]. They draw from the context of heat distribution in a building with multiple rooms, where each room is further divided into smaller offices. The walls between offices are poor insulators, allowing heat to spread quickly within a room, while the walls between rooms are better insulated, slowing heat transfer between them. Initially, large temperature differences may exist between offices, but within a short time, the temperature stabilizes within each room. However, significant temperature variations between rooms persist for a much longer period. Eventually, as heat gradually transfers between rooms, the entire building reaches thermal equilibrium. This process mirrors the behavior of nearly-completely decomposable Markov models. Fast processes, such as heat transfer within a room, stabilize quickly, while slower processes, such as heat transfer between rooms, continue to influence the system over a longer period. If a model focuses only on the short-term dynamics of the faster process and disregards the slower one, it may lead to significant inaccuracies in the long run.

There are several reasons why researchers are inclined to use decomposition-based approximations despite these risks. These methods offer faster computation and simplify model complexity, making them particularly attractive when computational resources are limited. In many cases, the primary interest may lie in the behavior of the faster process, leading to the assumption that slower processes can be neglected without introducing significant errors. Additionally, slow processes may be unknown at the modeling stage or difficult to incorporate into a mathematical framework. For example, in network modeling, service down times and scheduled maintenance do not always follow Markovian properties and are often excluded from the model to maintain mathematical tractability.

The results of this study emphasize the importance of a careful assessment of whether a given system is suitable for approximation. A large separation in time scales alone does not ensure accuracy. Instead, the structural properties of the system, such as the existence of a valid product-form decomposition, and the given prerequisites for an approximation approach, determine whether an approximation remains reliable. Models should be validated against real-world system behavior to ensure that omitting slow processes does not introduce significant errors.

ACKNOWLEDGMENTS

This work was partly funded by the Bavarian Ministry of Economic Affairs, Regional Development and Energy under grant number DIK0250/02 (project KOSINU5) and by the German Federal Ministry of Education and Research, Grant 18KIS2282 "SUSTAINET-Advance", sub-project 6G-ECONETS of the University of Würzburg "Energy-aware Optimization in Converged Networks for Sustainable 6G".

REFERENCES

[1] P. J. Courtois, *Decomposability: Queueing and Computer System Applications*. Academic Press, 1977.
[2] ——, "Error Analysis in Nearly-Completely Decomposable Stochastic Systems," *Econometrica: Journal of the Econom. Soc.*, 1975.
[3] C. Baur, "markov-approximation," Mar. 2025. [Online]. Available: https://github.com/lsinfo3/markov-approximation
[4] P. Tran-Gia and T. Hoßfeld, *Performance Modeling and Analysis of Communication Networks*. Würzburg University Press, 2021. [Online]. Available: https://modeling.systems
[5] J. G. Kemeny, J. L. Snell *et al.*, *Finite Markov Chains*, 1969.
[6] F. Metzger *et al.*, "Modeling of Aggregated IoT Traffic and its Application to an IoT Cloud," *IEEE*, 2019.
[7] F. Loh *et al.*, "Uplink vs. Downlink: Machine Learning-Based Quality Prediction for HTTP Adaptive Video Streaming," *Sensors*, 2021.
[8] R. E. Brown *et al.*, "Distribution System Reliability Assessment using Hierarchical Markov Modeling," *Transactions on power Delivery*, 1996.
[9] P. Buchholz, "Hierarchical Markovian Models: Symmetries and Reduction," *Performance Evaluation*, 1995.
[10] L. Muscariello *et al.*, "Markov Models of Internet Traffic and a New Hierarchical MMPP Model," *Computer communications*, 2005.
[11] L. Bright and P. G. Taylor, "Calculating the Equilibrium Distribution in Level Dependent Quasi-Birth-and-Death Processes," *Stochastic Models*, 1995.
[12] J. Kuntz, P. Thomas, G.-B. Stan, and M. Barahona, "Stationary Distributions of Continuous-Time Markov Chains: a Review of Theory and Truncation-based Approximations," *SIAM Review*, 2021.
[13] J. P. Kharoufeh, "Level-Dependent Quasi-Birth-and-Death Processes," 2011.
[14] H. Baumann and W. Sandmann, "Steady State Analysis of Level Dependent Quasi-Birth-and-Death Processes with Catastrophes," *Computers & operations research*, 2012.
[15] P. E. Heegaard and K. S. Trivedi, "Network Survivability Modeling," *Computer Networks*, 2009.
[16] A. K. Somani *et al.*, "An Efficient Decomposition Technique for Markov-Chain Analysis," in *Reliability and Maintainability Symp.*, 1995.
[17] M. Woodward, "Product Form Distributions from Decompositions of Markov Chains," *Electronics Letters*, 1994.
[18] H. A. Simon and A. Ando, "Aggregation of Variables in Dynamic Systems," *Econometrica: journal of the Econometric Society*, 1961.
[19] P. J. Schweitzer, "Aggregation Methods for Large Markov Chains," in *Workshop on Computer Performance and Reliability*, 1983.
[20] ——, "A Survey of Aggregation-Disaggregation in Large Markov Chains," *Numerical solution of Markov chains*, 1991.
[21] M. N. Katehakis and L. C. Smit, "A Successive Lumping Procedure for a Class of Markov Chains," *Probability in the Engineering and Informational Sciences*, 2012.
[22] P. Buchholz, "Exact and Ordinary Lumpability in Finite Markov Chains," *Journal of applied probability*, 1994.
[23] X. Gao *et al.*, "Investigating security vulnerabilities in a hot data center with reduced cooling redundancy," *IEEE Transactions on Dependable and Secure Computing*, 2020.