This chapter gives a brief overview over Markov models, a useful formalism to analyse stochastic systems.

8.1 Introduction to Markov Models

To estimate the dependability metrics of a system, knowledge about its behaviour is necessary. This behaviour can be measured during the operation of the system, resulting in a behaviour model (possibly without knowledge of the system’s interna). Alternatively, a model of the system can be created at design time to predict its behaviour, so that the internal structure of the system is reflected by the model.

The constructed system will execute in an environment where it will be subject to indeterministic influences (for example, the availability of services the system depends on might exhibit random behaviour). This means that the model must reflect the random/stochastic behaviour of the entire system and that the mathematical parameters must be determined at design time or identified at runtime.

In both scenarios, Markov models prove useful, since they are accompanied by algorithms that allow to compute relevant figures (metrics) or random distributions of those figures. For example, the behaviour of a software component with the identified states “working” ($S_w$) and “broken” ($S_b$) can be expressed by the Markov model shown in figure 1. Transitions denote possible state changes and are annotated with probabilities. With this Markov model, metrics such as Mean Time To Failure (MTTF) (see Sect. 9.3) and others can be computed.

![Graph and transition probabilities matrix $P$ for the Markov model of a simple software component](image)

The concepts and formulas that Markov models are built on will be introduced in the next sections in detail, accompanied by examples and the characteristics that can be obtained from the models. In figure 1, the Markov model is accompanied by the matrix $P$ that specifies its transition probabilities. For example, the probability to change from "working" state to "broken" state is 0.05.

The Markov model shown here is a so-called discrete-time Markov chain. It is assumed that the system’s state only changes after a fixed time interval. For example,
if the component state is determined every second and the component is definitely in working condition when started up \((t = 0)\), the initial state probability vector \(p_0\) (here, \(p_0 = (1 0)\)) can be used to start the second-after-second chain of states. By multiplying \(p_0\) with the matrix \(P\), \(p_1\) is obtained, which corresponds to the component’s state probabilities at time \(t = 1\): \(p_1 = (0.95 0.05)\).

The above example has an important property, the Markov property which describes the situation where the probability to be in a particular state of a model at time \(t + 1\) depends only on the state in which the model was at time \(t\), as well as on the state transition probabilities (matrix \(P\)) associated with the model. This independence of past states (at time \(t - 1, t - 2\) etc.) is sometimes referred to as memorylessness of a Markov model and has a large impact on the applicability of the model. The formal definition of the Markov property (not only for discrete time values as in the above example, but also for continuous time Markov models) will be given in the next sections.

A Markov chain consists of a set of discrete states and is in exactly one of these states at any point of time. Markov chains are widely used, for example in queuing theory (cf. Sect. 9.2). The notion of a chain was chosen because the Markov chain is a stochastic process that "walks" from one state to exactly one state at transition time, allowing for a sorted representation of the chain’s condition.

The transition probabilities characterise the Markov chain and allow for meaningful classification of both Markov chains and their single states. In some cases, the structure of the Markov model is unknown and must be reconstructed from data that has been collected empirically, for example in Hidden Markov Models (HMMs, see [323]).

We will start with the discussion of discrete-time Markov chains (DTMCs) in Sect. 8.2 and then generalise to continuous-time Markov chains (CTMCs) in Sect. 8.3 before presenting applications of Markov models for dependability analysis in Sect. 8.4. Markov Reward Models (MRMs) are treated separately in Sect. 24.2 of Chap. 24.2.

### 8.2 Discrete-Time Markov Chains - DTMC

#### Overview

Formally, a discrete-time Markov chain (DTMC) is defined as a stochastic process with the discrete parameter space \(T := \{0, 1, 2, \ldots\}\). The (discrete) state space \(I\) of a DTMC consists of random variables \(X_t\) with \(t \in T\) \((X_0 = s_0\) is the starting state) and permits to state the Markov property for DTMCs as

\[
P(X_n = s_n | X_0 = s_0, X_1 = s_1, \ldots, X_{n-1} = s_{n-1}) = P(X_n = s_n | X_{n-1} = s_{n-1})
\]

From now on, a shorter notation for probability mass functions will be used:

\(p_{ij}(m, n) := P(X_n = j | X_m = i)\) and \(p_i(n) := P(X_n = i)\)

We will limit ourselves to homogeneous Markov chains: for them, the transition probability from state \(j\) at time \(m\) to state \(k\) at time \(n\) only depends on the "distance" \(n - m\) between both states and is independent of actual values of \(m\) or \(n\).

---

1 The more general notion of Markov processes is not limited to discrete states but will not be considered here.
Formally, for homogeneous DTMCs, \( p_{ij}(d) := p_{i,j}(t, t + d) \forall t \in T, d \geq 0 \).

The above software component example is a homogeneous discrete time Markov chain, since the transition probabilities are independent of the time.

Using the homogeneous one-step transition probabilities \( p_{ij}(1) \), it is straightforward to compute the probability of the chain \( s_0, s_1, \ldots, s_n \): its probability is

\[
p_{s_0} \cdot p_{s_0s_1}(1) \cdot p_{s_1s_2}(1) \cdot \ldots \cdot p_{s_{n-1}s_n}(1)
\]

(where \( p_{s_0} \) is the probability to be in initial state \( s_0 \)). The one-step transition probabilities are collected in the transition probability matrix \( P := [p_{ij}(1)] \) for which \( 0 \leq p_{ij}(1) \leq 1 \) and \( \sum_{k \in I} p_{ij}(1) = 1 \) hold, \( \forall j \in I \). The graph in figure 1 is called the state transition diagram.

To obtain the \( n \)-step transition probabilities \( p_{ij}(n) \) for any \( n \) other than 1, we define \( p_{ij}(0) \) to be 1 if \( i = j \) and 0 otherwise. Then, using known \( p_{ij}(1) \), we can generalise to any \( n \) through the Chapman-Kolmogorov equation if the length \( n \) of a transition is split into two parts (of length \( l \) and \( m \)) and all possible states reached after \( l \) transitions are considered.

\( p_{ij}(n) \) is then seen as the sum of probabilities of "ways" that pass through all possible states \( j \) after \( l \) transitions: with \( l + m = n, l > 1, \) \( m > 1, \)

\[
p_{ik}(l + m) = \sum_j p_{ij}(l)p_{jk}(m)
\]

It can be proved that the calculation of \( n \)-step transition probabilities can equivalently be done by obtaining the \( n \)-step stochastic matrix \( P(n) \) from the one-step matrix \( P \) with the formula \( P(n) = P^n = P \cdot P(n-1) \). If the probabilities for states 0, 1, \ldots at time \( t \) are collected into a vector \( p(t) := [p_0(t), p_1(t), \ldots] \), the formula to compute \( p(n) \) using the matrix \( P \) is

\[
p(n) = p(0) \cdot P(n) = p(0) \cdot P^n
\]

For the software component example (which had \( P = \begin{pmatrix} 0.95 & 0.05 \\ 0.80 & 0.20 \end{pmatrix} \) and \( p(0) = [1, 0] \)), the computation of \( p(2) \) yields

\[
p(2) = [1, 0] \cdot \begin{pmatrix} 0.95 & 0.05 \\ 0.80 & 0.20 \end{pmatrix}^2 = [1, 0] \cdot \begin{pmatrix} 0.9425 & 0.0575 \\ 0.9200 & 0.0800 \end{pmatrix} = [0.9425, 0.0575]
\]

**State Classification**

States are classified according to the number of visits to detect tendencies and specifics of a model. To illustrate the state classification with an appropriate example, we will extend our previous example with additional states, as shown in figure 2.

It is obvious that for the state \( S_i \) (where the component is initialised), the Markov process will not return to it, since no transitions to \( S_i \) exist. Such states of a DTMC are called *transient* or, equivalently, *nonrecurrent*. A state that cannot be left once a Markov process has entered it is called *absorbing* and can be recognised by \( p_{ii}(1) = 1 \) (there is no such state in figure 2).
States $S_b$ ("broken") and $S_w$ ("working") are recurrent since the process will eventually return to them with probability 1 after some unspecified time $t$. These two states communicate since there are directed paths from $S_b$ to $S_w$ and vice versa (in contrast, $S_i$ and $S_w$ do not communicate).

For recurrent states, one interesting measure is the mean recurrence time, which can be used for the mean time to failure (MTTF) metric. With $f_{ii}(n)$ as the probability that the process will return to state $i$ after exactly $n$ steps for the first time, the mean recurrence time $\mu_i$ for state $i$ can be computed to be

$$\mu_i = \sum_{n=1}^{\infty} nf_{ii}(n)$$

Depending on the value of $\mu_i$, a state is recurrent nonnull/positive recurrent if $\mu_i$ is finite and recurrent null if $\mu_i$ is infinite. It is easy to see that for any positive recurrent state $i$, $\exists K \forall k > K : f_{ii}(k) = 0$, $f_{ii}(K) > 0$, i.e. the state will always be revisited after $K$ or less steps.

For any recurrent state, its period, defined as the greatest common divisor of all $n > 0$ for which $p_{ii}(n) > 0$, can be computed. If the period is greater than 1, the state is called periodic, and aperiodic otherwise. The states $S_b$ and $S_w$ are both aperiodic; state $S_i$ is transient and thus neither aperiodic nor periodic. The overall state classification of discrete-time Markov chains as outlined in [472] is displayed in figure 3.

8.3 Continuous-Time Markov Chains (CTMC)

Mathematical Foundations

A Continuous-Time Markov Chain (CTMC) allows state changes at any instance of time, leading to continuous parameter space $T := [0, \infty)$, but state space $I$ remains discrete as in DTMCs. Reformulation of the Markov property for CTMC, given an increasing parameter sequence $0 \leq t_0 < t_1 < \ldots < t_{n-1} < t_n$, yields the requirement that

$$P(X(t_n) = x_n | X(t_{n-1}) = x_{n-1}, \ldots, X(t_0) = x_0) = P(X(t_n) = x_n | X(t_{n-1}) = x_{n-1})$$
As in the previous section, two structures completely define a Markov chain:

1. the vector \( P(X_{t_0}) = \{k|k = 0, 1, 2, \ldots\} \) that contains the initial probabilities \( P(X_{t_0} = k) \) for all states \( k \) at time \( t_0 \)
2. the transition probabilities

\[
p_{ij}(t, v) = P(X(v) = j|X(t) = i)
\]

with \( 0 \leq t \leq v \) and the special cases \( p_{ii}(t, t) = 1 \) and \( p_{ij}(t, t) = 0 \) for \( i \neq j \)

Note that \( \forall i, 0 \leq t \leq v \) the transition probabilities fulfill \( \sum_{j \in I} p_{ij}(t, v) = 1 \).

A CTMC is homogeneous with respect to time if for \( p_{ij}(t, v) \), only the time difference \( v - t \) matters, that is, if \( \forall d > 0, p_{ij}(t, v) = p_{ij}(t + d, v + d) \). For a homogeneous CTMC and time difference (distance) \( \tau \), we define

\[
\forall t \geq 0 : p_{ij}(\tau) := p_{ij}(t, t + \tau) = P(X(t + \tau) = j|X(t) = i)
\]

The general probability to be in state \( j \) at a time \( t \) is defined as

\[
\pi_j(t) := P(X(t) = j)
\]

with

\[
P(X(t) = j) = \sum_{i \in I} P(X(t) = j|X(v) = i)P(X(v) = i) = \sum_{i \in I} p_{ij}(o, t)\pi_i(0)
\]

Since it is difficult to work with the dynamic Chapman-Kolmogorov equation

\[
p_{ij}(t, v) = \sum_{k \in I} p_{ik}(t, u)p_{kj}(u, v) \quad (0 \leq v < u < t)
\]

the rates of transitions for CTMC are defined as follows: net rate out of state \( j \) at time \( t \) is

\[
q_j(t) := q_{jj}(t) = -\frac{d}{dt}p_{jj}(v, t)|_{v=t} = \lim_{h \to 0} \frac{p_{jj}(t, t+h) - p_{jj}(t, t)}{h} = \lim_{h \to 0} \frac{1-p_{jj}(t, t+h)}{h}
\]

and the rate from state \( i \) to state \( j \) at time \( t \) (for \( i \neq j \)) is

\[
q_{ij}(t) := \frac{d}{dt}p_{ij}(v, t)|_{v=t} = \lim_{h \to 0} \frac{p_{ij}(t, t+h) - p_{ij}(t, t)}{h} = \lim_{h \to 0} \frac{p_{ij}(t, t+h)}{h}
\]
\( q_j(t) \) is the net rate out of state \( j \) at time \( t \), while \( q_{ij}(t) \) is the rate from state \( i \) to state \( j \) at time \( t \). Over a series of transformations (cf. [472]), the Kolmogorov Differential Equations can be obtained (with \( Q(t) := [q_{ij}(t)] \) and \( q_{ii}(t) = -q_i(t) \)):

\[
\frac{dP(v, t)}{dt} = P(v, t)Q(t) \quad \text{and} \quad \frac{dP(v, t)}{dv} = Q(t)P(v, t)
\]

Rewriting this equations for a homogeneous CTMC, we obtain

\[
\frac{dP(t)}{dt} = P(t)Q
\]

and

\[
\frac{d\pi(t)}{dt} = \pi(t)Q
\]

The (infinitesimal) generator matrix \( Q := [q_{ij}(t)] \) has the following properties: row sums are all equal to 1, \( \forall i : q_{ii} \leq 0 \) and the transition rates from state \( i \) to state \( j \) fulfill \( \forall i \neq j, q_{ij} \geq 0 \). For the discussion of semi-Markov processes as well as non-homogeneous CTMCs, we point to [472].

**State Classification**

While recurrent null/non-null and transient states have the same meaning in CTMCs as in DTMCs, a CMTC state \( i \) is absorbing if \( \forall t \forall j \neq i \) holds \( p_{ij}(t) = 0 \). A CTMC is irreducible if \( \forall i, j \exists t \) so that \( p_{ij}(t) > 0 \) (i.e., any state can be reached from any other state in a transition of suitable duration \( t \) with non-zero probability).

**Steady-state values**

\( \pi := \{\pi_j | j = 0, 1, 2, \ldots\} \)

of a CTMC are defined by

\[ \pi_j := \lim_{t \to \infty} \pi_j(t) \]

and are subject to \( \sum_j \pi_j = 1 \) and to \( \pi Q = 0 \). An irreducible CTMC with all states being recurrent non-null will have unique \( \pi \) that do not depend on the initial probability vector; all states of a finite irreducible CTMC are recurrent non-null, so it is possible to compute the steady-state values for it.

**Queues and Birth-Death Processes**

Although the equations of continuous-time Markov chains look intimidating at first sight, they have a strong practical importance as the queuing theory is footed on them. We will introduce queues starting with the example from figure[4] and also briefly discuss birth-death processes which represent the abstract structures behind the queues.

The pictured queue shall have the properties that elements to it arrive one-by-one in a random manner and that there is only one "server position" where elements are processed on FCFS basis, after which the elements leave the queue. The elements arrive at the end of the queue with the birth rate \( \lambda \); after being served, the elements leave the queue with the death rate \( \mu \). Equivalently, the arrival times are exponentially distributed
with mean $1/\lambda$ and the service times are exponentially distributed with mean $1/\mu$. In figure\textsuperscript{5} the transition state diagram for this queue is pictured.

One of the quantities that a queuing model can help to determine the expected (average) number of elements in the system, $N(t)$, using the traffic intensity $\rho$

$$\rho := \frac{\lambda}{\mu}$$

of the queue. Considering the possible numbers $k$ of elements in the system ($k \in [0, \infty)$, $k \in \mathbb{N}$), we need their probabilities $\pi_k$ for weighting:

$$E[N] = \sum_{k=0}^{\infty} k \pi_k$$

Without citing the proof (which can be found in \textsuperscript{472}), following formulas can be used if $\rho < 1$ (i.e. if the queue is stable) to compute the steady-state probabilities $\pi_i$ for any state $i$:

$$\pi_0 = 1 - \frac{\lambda}{\mu} = 1 - \rho \quad \text{and} \quad (\text{for } k \geq 1) \quad \pi_k = \left(\frac{\lambda}{\mu}\right)^k \pi_0 = \rho^k \pi_0 = \rho^k (1 - \rho)$$

As an example, consider a software component that compresses chunks of data with various sizes. The component is implemented as a single thread and sends/receives the data chunks asynchronously. The time between data chunks arrivals is exponentially distributed with mean 100ms ($= 1/\lambda$). Since the sizes of data chunks vary, the service time is also exponentially distributed with the mean 75ms ($= 1/\mu$). Thus, the traffic intensity $\rho$ is $0.75 < 1$ and $\pi_0 = 0.25$, $\pi_1 = 0.1875$, $\pi_2 = 0.140625$ and so on.

Using $\rho$ and given queue is in stable case, the mean and the variance of $N(t)$ can be computed without prior computation of all $\pi_k$ (see \textsuperscript{472}):

$$E[N] = \sum_{k=0}^{\infty} k \pi_k = \frac{\rho}{1 - \rho} \quad \text{and} \quad Var[N] = \frac{\rho}{(1 - \rho)^2}$$

For the above data compression example, $E[N] = 3$ and $Var[N] = 12$. 

---

**Fig. 4.** A simple queue (M/M/1) as an example of a continuous-time Markov chain

**Fig. 5.** The transition state diagram for the queue from figure\textsuperscript{4}
Another measure of interest in queues is the response time $R$, i.e. between the arrival of the element at the end of the queue and the time that element leaves the queue. To compute the mean of this random variable in the steady state case ($\rho < 1$), the Little’s formula $E[N] = \lambda E[R]$ is used, as it creates the relation between the response time and the number of elements in the queue. Its interpretation is that the (mean) number of elements in the queue is the product of arrival rate and the (mean) response time. Using the above formula for $E[N]$,

$$E[R] = \sum_{k=0}^{\infty} \frac{k}{\lambda} \pi_k = \frac{1/\mu}{1 - \rho}$$

can be derived. $\frac{1/\mu}{1 - \rho}$ can be interpreted as the quotient of the (mean) service time and the probability of the queue to be empty. For the above example, $E[R] = 300$ms.

### 8.4 Markov Chains and Dependability Analysis

Markov chains will be used on several occasions throughout this volume. In Sect. 9.3, Markov process theory is used to calculate hardware availability through employment of continuous-time stochastic logic (CSL) and with possible application of model-checking algorithms.

In Sect. 9.4, assessing hardware reliability for complex, fault-tolerant systems with built-in redundancy is done, among other approaches, with Markov chains. Markov chains also permit to model systems that work if $m$ or less out of $n$ components have failed (with identical or distinct components as well as same or different failure/repair rates), and provide algorithms to compute the steady state probabilities of the parametrised Markov chains.

On the other hand, Markov chains can be used for modeling software reliability, especially for component-based systems (cf. Sect. 10.4). This approach is particularly suitable for reliability prediction during the design phase, even before a "black box" models of real components or software systems become available.

Markov models serve as the foundation for Markov Reward Models (MRMs), which are used in Sect. 24.2 to measure the effects of a system’s performance degradation. This is done in a larger context of performability, i.e. the combination of performance and reliability and allows to reward the system for the time spent in states that representing readiness of the system.