Chapter 5
Phase-Type Distributions

Philipp Reinecke, Levente Bodrog and Alexandra Danilkina

Abstract Both analytical (Chap. 6) and simulation- and experimentation-based (Chap. 17) approaches to resilience assessment rely on models for the various phenomena that may affect the system under study. These models must be both accurate, in that they reflect the phenomenon well, and suitable for the chosen approach. Analytical methods require models that are analytically tractable, while methods for experimentation, such as fault-injection (see Chap. 13), require the efficient generation of random-variates from the models. Phase-type (PH) distributions are a versatile tool for modelling a wide range of real-world phenomena. These distributions can capture many important aspects of measurement data, while retaining analytical tractability and efficient random-variate generation. This chapter provides an introduction to the use of PH distributions in resilience assessment. The chapter starts with a discussion of the mathematical basics. We then describe tools for fitting PH distributions to measurement data, before illustrating application of PH distributions in analysis and in random-variate generation.

P. Reinecke (✉) · A. Danilkina
Institute of Computer Science,
Free University Berlin,
Takustr. 9, 14195 Berlin, Germany
e-mail: philipp.reinecke@fu-berlin.de

L. Bodrog
Department of Telecommunications,
Budapest University of Technology and Economics,
Budapest 1521, Hungary
e-mail: bodrog@webspn.hit.bme.hu

A. Danilkina
e-mail: danilkin@zedat.fu-berlin.de

K. Wolter et al. (eds.), *Resilience Assessment and Evaluation of Computing Systems*,
DOI: 10.1007/978-3-642-29032-9_5, © Springer-Verlag Berlin Heidelberg 2012

5.1 Introduction

Phase-type (PH) distributions are an often-used type of model for many phenomena in system evaluation, e.g., service-times, delays, and failure times. This chapter provides a gentle introduction to the theory of PH distributions and their application in common evaluation tasks.

As an illustrative example we consider resilience evaluation of a simple system where clients are being served by a faulty server. The server uses multiple threads that require access to shared resources, but resource contention may lead to a deadlock, which manifests as a situation where no service is provided anymore. We assume that we do not have the means to address the root of the problem in the server itself, but might be able to reset the server once it has reached a deadlock. In order to assess resilience of a system relying on this server to work, we want to model the time between server crashes. We use PH distributions for this task, as they provide good approximation of the time-to-failure distribution and are well-suited for both analytical approaches and simulation.

The general workflow for applying phase-type distributions in evaluation tasks is shown in Fig. 5.1: first, a PH distribution describing the phenomenon under study has to be found. This can be achieved both with a white-box and a black-box approach. With the white-box approach (Sect. 5.3), the structure of the system is used to directly infer a PH distribution that describes the behaviour of the system. With the black-box approach (Sect. 5.4), system behaviour is measured and the measurements are fitted by a PH distribution. Both approaches result in a distribution that is a model for the behaviour of the system. This distribution can then be used in analytical approaches such as matrix-analytic methods (Sect. 5.5) and in simulation (Sect. 5.6).

We are going to show how to arrive at a PH distribution for the time-to-failure distribution for our example system using both the white-box and the black-box approach. We will then illustrate application of the distribution both in analytical and in simulation approaches. First, however, we need to introduce the required mathematical background and notation.

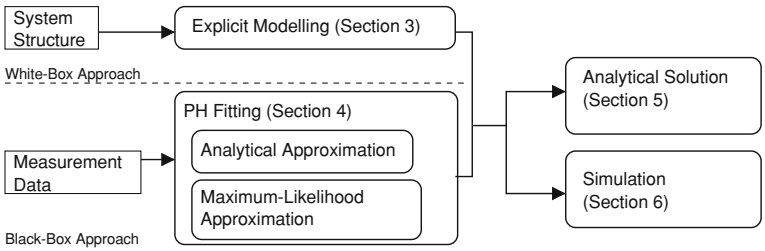

Fig. 5.1 Typical workflow when applying phase-type distributions in system evaluation

5.2 Mathematical Background

Continuous phase-type (PH) distributions represent the time to absorption in a Continuous-Time Markov Chain (CTMC) with one absorbing state [683]. PH distributions are commonly specified by a vector-matrix tuple $(\boldsymbol{\alpha}, \mathbf{A})$, where

$$\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n) \in \mathbf{R}^n \quad \text{and} \quad \mathbf{A} = \begin{pmatrix} \lambda_{11} & \cdots & \lambda_{1n} \\ \vdots & \ddots & \vdots \\ \lambda_{n1} & \cdots & \lambda_{nn} \end{pmatrix} \in \mathbf{R}^{n \times n}.$$

Definition 5.1 The *size* of the $(\boldsymbol{\alpha}, \mathbf{A})$ representation is the size of the vector $\boldsymbol{\alpha}$, which is equal to the size of the square matrix $\mathbf{A}$.

Definition 5.2 The *probability density function (PDF)*, *cumulative distribution function (CDF)*, *Laplace-Stieltjes Transform (LST)* of the CDF and kth *moment*, respectively, are defined as follows [446, 683, 870]:

$$f(x) = \boldsymbol{\alpha} \mathrm{e}^{\mathbf{A}x} \mathbf{a}, \tag{5.1}$$

$$F(x) = 1 - \boldsymbol{\alpha} \mathrm{e}^{\mathbf{A}x} \mathbb{1}, \tag{5.2}$$

$$\tilde{F}(s) = \alpha_{n+1} + \boldsymbol{\alpha}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{a}, \tag{5.3}$$

$$E\left[X^k\right] = k!\boldsymbol{\alpha}(-\mathbf{A})^{-k}\mathbb{1}. \tag{5.4}$$

where $\mathbb{1}$ is the column vector of ones of the appropriate size and $\mathbf{a} = -\mathbf{A}\mathbb{1}$. Note that, since $\boldsymbol{\alpha}$ is a row vector and both $\mathbb{1}$ and $\mathbf{a}$ are column vectors, the above equations do indeed specify scalar values. Furthermore, observe that phase-type distributions have rational LST and that the eigenvalues of the transient generator matrix are the poles of the LST of the distribution [697].

The vector-matrix representation of a PH distribution is not unique. In general, there exists another representation $(\boldsymbol{\alpha}', \mathbf{A}')$ of size m that represents the same phase-type distribution. Different representations of a PH distribution may differ both in size $(n \neq m)$ and in the contents of the tuples.

Another representation of the same size can be computed by a similarity transformation, as follows: when $\mathbf{B}$ is invertible and $\mathbf{B}\mathbb{1} = \mathbb{1}$, then $(\boldsymbol{\alpha}\mathbf{B}, \mathbf{B}^{-1}\mathbf{A}\mathbf{B})$ is another representation of the same distribution, since its CDF is

$$1 - \boldsymbol{\alpha}\mathbf{B}\mathrm{e}^{\mathbf{B}^{-1}\mathbf{A}\mathbf{B}x}\mathbb{1} = 1 - \boldsymbol{\alpha}\mathbf{B}\mathbf{B}^{-1}\mathrm{e}^{\mathbf{A}x}\mathbf{B}\mathbb{1} = 1 - \boldsymbol{\alpha}\mathrm{e}^{\mathbf{A}x}\mathbb{1}.$$

It is also possible to generate representations of the same distribution with another size, using a non-square matrix $\mathbf{W}$.

An important property of PH distributions (and in fact one which distinguishes them from larger classes such as the Matrix-Exponential (ME) distributions) is that every PH distribution has a *Markovian* representation $(\boldsymbol{\alpha}, \mathbf{A})$. This representation

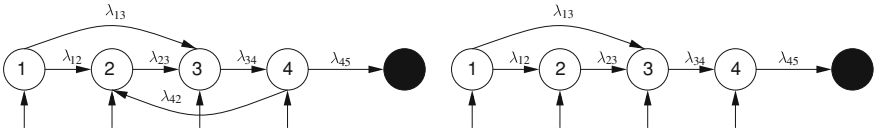

Fig. 5.2 CTMC representations for general and acyclic phase-type distributions

admits an interpretation of the PH distribution as the distribution of absorption-times in a Markov chain. With the Markovian representation, $\mathbf{A}$ describes the transient part of the generator matrix of the associated CTMC,

$$\overline{\mathbf{A}} = \begin{pmatrix} \mathbf{A} & \mathbf{a} \\ \mathbf{0} & 0 \end{pmatrix},$$

and consequently fulfills the required properties: all off-diagonal elements are non-negative ($\lambda_{ij} \geq 0 \, (1 \leq i \neq j \leq n)$), all diagonal elements are negative, and the row-sums are non-negative ($\mathbf{a} = -\mathbf{A}\mathbb{1} \geq 0$). The vector $\boldsymbol{\alpha}$ is the vector of initial probabilities of the transient states of the CTMC, and thus $\boldsymbol{\alpha} \geq 0$ and $\boldsymbol{\alpha}\mathbb{1} \leq 1$. In the following, we focus on the Markovian representation of phase-type distributions, where we assume that $\boldsymbol{\alpha}\mathbb{1} = 1$, i.e., there is no probability mass at zero.

5.2.1 PH Classes

Based on the structure of the underlying Markov chain, several classes of phase-type distributions can be distinguished. These classes differ in the statistical properties they can represent. Furthermore, the structure of a PH representation often has an impact on its application, as some structures allow more efficient solutions.

The most important distinction is the one into Acyclic and General Phase-type distributions: every acyclic phase-type (APH) distribution has at least one Markovian representation without cycles in the sub-generator, while for general phase-type distributions cycles are allowed. This is illustrated in Fig. 5.2: the distribution on the left contains a cycle, that is, a backward transition from state 4 to state 2. The distribution on the right does not contain this transition and therefore there are no cycles.

Most approaches in fitting and application of PH distributions focus on the APH class, as this class offers better tractability than the general PH class. Within APH, we distinguish two important sub-classes: the first one is the class of Hyper-Erlang distributions (HErD). Hyper-Erlang distributions are mixtures of Erlang-distributions with different lengths and rates. They can be specified by a tuple $(\boldsymbol{\beta}, m, \mathbf{b}, \boldsymbol{\lambda})$, where $\boldsymbol{\beta}$ is the vector of initial probabilities of each Erlang branch, m is the number of Erlang branches, $\mathbf{b}$ is the vector of the lengths of the Erlang branches, and $\boldsymbol{\lambda}$ is a vector containing the rates. The size of a Hyper-Erlang distribution is given by the

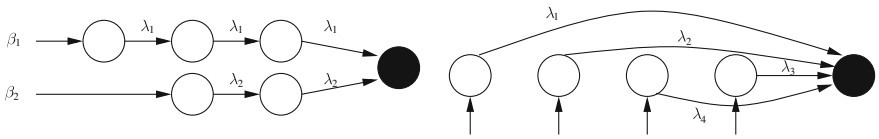

Fig. 5.3 CTMC representations for Hyper-Erlang and hyper-exponential distributions

sum of the lengths of the branches, i.e., $n = \mathbf{b}\mathbb{1}$. The general structure is illustrated in Fig. 5.3, where we show a hyper-Erlang distribution with $m = 2$ branches of length $b_1 = 3$ and $b_2 = 2$, respectively. The initial probabilities and the transition rates are given by $\boldsymbol{\beta} = (\beta_1, \beta_2)$ and $\boldsymbol{\lambda} = (\lambda_1, \lambda_2)$. The size of this representation is $n = b_1 + b_2 = 5$. One important example is the Erlang distribution, i.e., a Hyper-Erlang distribution with only one branch and initial probability $\beta_1 = 1$.

The second sub-class of APH we consider is the class of Hyper-Exponential distributions (HEx) of order n, specified by initial probability vector $\boldsymbol{\alpha}$ and rate vector $\boldsymbol{\lambda}$. Figure 5.3 shows an example for a hyper-exponential distribution of size $n = 4$. From this example, it is obvious that the hyper-exponential distributions are a subclass of the hyper-Erlang distributions, as every hyper-exponential distribution is a hyper-Erlang distribution with branch length vector $\mathbf{b} = \mathbb{1}$. Furthermore, setting $n = 1$ and $\alpha_1 = 1$ yields the exponential distribution with rate λ_1.

5.2.2 Canonical Representations

While in general representations for phase-type distributions are not unique, several canonical forms have been defined. For each PH distribution, the canonical form of a given size n is unique in the sense that there exists no representation of the same size n with the structure of the canonical form, but different parameters. Therefore, by comparing canonical forms, we can determine whether PH distributions given by different representations are identical. More important, however, is the use of canonical forms in fitting, analysis, and simulation, where their typically low number of parameters and simple structure enable efficient methods.

In the following we discuss Cumani's Canonical Form 1 (CF-1) [251] and the Monocyclic form introduced in [659], as these are the most common ones.

5.2.2.1 The Canonical Form for APH Distributions

The Canonical Form 1 (CF-1) was defined in [251]. The structure of its underlying CTMC is shown in Fig. 5.4: the Markov chain can be entered at any state $i = 1, \ldots, n$ with probability α_i, but the absorbing state can only be reached by traversing all remaining states. For this structure, the associated generator $\mathbf{A}$ has a bi-diagonal structure, that is, for $i = 1, \ldots, n-1$ and $1 \leq j \leq n$,

$$-\lambda_{ii} = \lambda_{i,i+1} \quad \text{and} \quad \lambda_{ij} = 0 \text{ for } j \notin i, i+1.$$

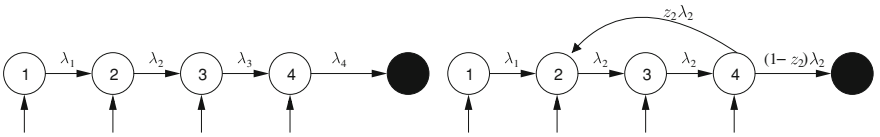

Fig. 5.4 Canonical representations for phase-type distributions

It is often convenient to describe a bi-diagonal generator by the vector

$$\boldsymbol{\Lambda} = (\lambda_1, \ldots, \lambda_n),$$

where $\lambda_i = |\lambda_{ii}|$ for $i = 1, \ldots, n$. The formal definition for the CF-1 form is then

Definition 5.3 [251] The *Canonical Form 1 (CF-1 form)* is a bi-diagonal Markovian representation $(\boldsymbol{\alpha}, \boldsymbol{\Lambda})$ where the elements of the diagonal, given in the vector $\boldsymbol{\Lambda}$, are ordered by absolute value.

In [251, 698] it has been shown that every acyclic phase-type distribution with a Markovian representation of size n has a unique CF-1 representation of the same size.[1] The CF-1 form for an APH given as $(\boldsymbol{\alpha}, \mathbf{A})$ can be obtained by a similarity transformation. A procedure for constructing the similarity transformation matrix is given in [422].

Note that transforming an APH representation of size n to the CF-1 form considerably reduces the number of parameters: a general APH representation has n initial probabilities $\alpha_1, \ldots, \alpha_n$ and n^2 entries in the subgenerator matrix $\mathbf{A}$, i.e., the number of parameters is $n + n^2$. In the CF-1 form $\mathbf{A}$ is an upper bi-diagonal matrix with $\lambda_{ii} = -\lambda_{i+1,i}$. The CF-1 form therefore only requires the n rates on the diagonal and the n entries in $\boldsymbol{\alpha}$, resulting in $2n$ parameters.

5.2.2.2 The Monocyclic Form for General PH Distributions

General PH distributions may have complex poles, and the poles of a PH distribution are given by the eigenvalues of the subgenerator matrix $\mathbf{A}$. As the eigenvalues of a bi-diagonal representation $(\boldsymbol{\alpha}, \mathbf{A})$ are equal to entries of the diagonal and $\mathbf{A} \in \mathbf{R}^{n \times n}$ it is easy to see that a bi-diagonal structure like the CF-1 form cannot represent phase-type distributions with complex poles.

For this reason, [659] proposed the Monocyclic form as a chain of Feedback-Erlang (FE) blocks, defined as follows:

Definition 5.4 A *Feedback-Erlang (FE) block* is given by a tuple (b, λ, z) of the length b, transition rate λ, and feedback probability $z \in [0, 1)$. The Feedback-Erlang

[1] Smaller CF-1 representations may exist if there is redundancy in the original representation [422, 683, 750].

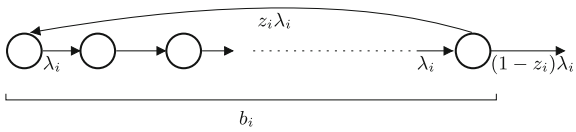

Fig. 5.5 Structure of a Feedback-Erlang block

block consists of an Erlang-distribution with length b and rate λ and an additional (feedback) transition from the last state of the block to the first state.

Figure 5.5 illustrates this concept. Note that the cases $z = 0$ and $b = 1$ are allowed. For $z = 0$, the Feedback-Erlang is simply an Erlang of order b, while for $b = 1$ it is an exponential distribution. The importance of this structure lies in the fact that for $z > 0$ and $b > 1$ the block has a conjugate-complex pair of eigenvalues [659]. Therefore, a chain of FE blocks can be used to represent the complex eigenvalue pairs of a general phase-type distribution.

Based on this observation, [659] define the Monocyclic representation as a chain of Feedback-Erlang blocks:

Definition 5.5 A *Monocyclic representation* is given by the tuple $(\boldsymbol{\alpha}, m, \mathbf{b}, \boldsymbol{\lambda}, \boldsymbol{z})$, where the vector $\boldsymbol{\alpha} \in \mathbf{R}^{\mathbf{b}\mathbb{1}}$ specifies the initial state probabilities, and $\mathbf{b}$, $\boldsymbol{\lambda}$ and $\boldsymbol{z}$ define the length, rate, and feedback probability of the m Feedback-Erlang blocks.

The FE blocks are positioned such that the absolute values of the dominant eigenvalues r_i are in ascending order: $r_i \leq r_{i+1}$.

Any PH distribution has a Monocyclic representation [659]. If the representation of the PH distribution is PH-simple [698] and of size n, then the size of the Monocyclic representation is $n' \geq n$. This potential size expansion makes the Monocyclic representation less efficient in analytical studies, but its simple and still Markovian structure makes it promising for simulation studies.

The structure of a Monocyclic representation is shown in Fig. 5.4. Note that if $z_i = 0$ for all FE blocks $i = 1, \ldots, b$ the Monocyclic form is equivalent to the CF-1 form. That is, the CF-1 form is actually a special case of the Monocyclic form.

5.2.3 Properties

Phase-type distributions exhibit a number of properties that make them attractive for use in resilience evaluation. In particular, the support of PH distributions is the set of non-negative real numbers. Therefore, they can be used to model typical system properties such as response-times, interarrival times, or inter-failure times. Furthermore, the PH class is closed under important operations such as minimum, maximum, and summation, i.e., PH distributed random variables can be combined without losing the properties of PH distributions.

On the other hand, even though PH distributions are well-suited for fitting data (see Sect. 5.4), two important limitations may affect their suitability for particular

tasks. First, the density of a phase-type distribution is strictly positive [683]:

$$f(t) > 0, \quad t > 0,$$

i.e., phase-type distributions can only approximate the density if it is close to zero, and may require a large number of phases to do so, rendering models more complex. Second, PH distributions are limited with respect to the moments they can express. In particular, the feasible range of the squared coefficient of variation for the PH of size n (PH(n)) is

$$cv^2 \geq \frac{1}{n}, \tag{5.5}$$

where the equality holds for the Erlang distribution with n phases (Erl(n)) [260]. This bound implies that in order to approximate data with low variation a large number of phases is required.

For higher moments there is no general knowledge, however there are several special cases for which some insights on the moment bounds exist, like e.g., the moment bounds of the APH(2) $\equiv$ PH (2) class [871] and the moment bounds of the PH (3) class implied by the canonical form given in [447]. The bounds of the general APH class within the PH class are known according to the APH canonical form and there also exists a numerical method to determine the general PH bound in [448].

From the fitting perspective the *reduced moment problem* (when a distribution function is determined based on its moments) can also be crucial. This problem is only solved for the larger class of matrix-exponential distributions [907].

5.3 Explicit Modelling

We will now describe how a phase-type distribution can be obtained directly from the structure of the system under study. Returning to our example with the faulty server, recall that the server becomes unavailable due to resource contention between its threads. A very intuitive way of thinking about resource contention is in terms of the Dining Philosopher's problem [292][2]: at least two philosophers sitting around a table want to eat a dish for which they require a fork in each hand. However, there are only as many forks as philosophers. Each philosopher employs the following strategy: if a fork is available either to the left or to the right, they wait a random amount of time before taking it. Once they have one fork, they wait for the other one, before they start eating. They eat for a random amount of time and then drop both forks at the same time. It is immediately obvious that this strategy eventually leads to the situation where each philosopher has one fork in his left (or right) hand

[2] Note that we only use the general problem, given in [292], but do not assume a solution.

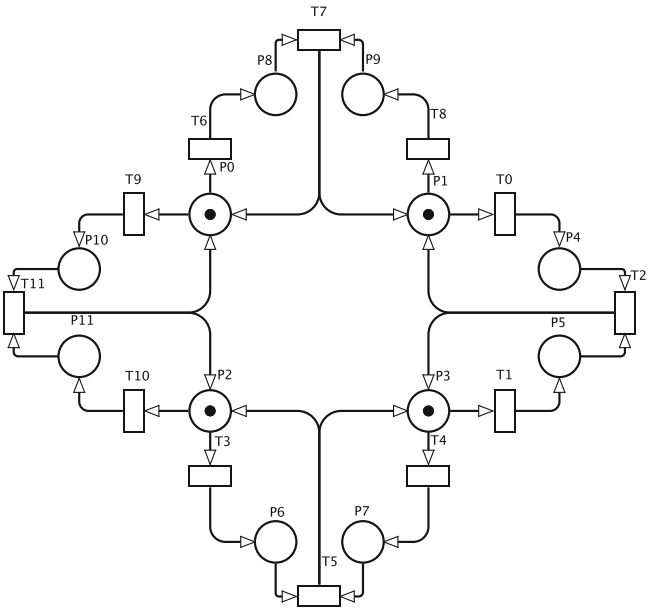

Fig. 5.6 Stochastic Petri Net (SPN) model for the Dining Philosopher's problem with four philosophers

and is waiting for the other one, which, however, is in the corresponding hand of his neighbour.

In terms of the server system, the philosophers represent the threads of the server and the forks are the shared resources. If each thread has access to one of the resources, but not to the other (each philosopher has one fork), the system is in the deadlock situation and unable to serve clients (no philosopher can eat). The only way of leaving this situation is to reset the system to the inital state where all resources are free (all forks are on the table).

Based on the abstraction as a Dining Philospher's problem, we can model our faulty server using a Stochastic Petri Net (SPN), as shown in Fig. 5.6 for $n = 4$: a marking on one of the four innermost places represents the respective fork resting on the table. The eight outermost places represent the hands of the philosophers, i.e., a marking in one of these places models that the philosopher has picked up a fork in this hand. The transitions leading from the inner places to the outer places model the act of picking up a fork with that hand, while the transitions from the outside to the inside model the laying down of both forks. We assume that both the times before picking up a fork and the eating time have exponential distribution, i.e., the transitions are Markovian.

The CTMC underlying this SPN possesses two absorbing states (the two deadlock situations). Since in both cases no philosopher can eat, or, equivalently, the server system cannot serve clients, both states describe the same situation and can be lumped

into one absorbing state. Consequently, the time to absorption, i.e., the time to failure, follows a phase-type (PH) distribution.

If we know the structure of the system and can build a CTMC model for it, using e.g., an SPN, we can thus directly derive a PH distribution. This distribution can then be used in further evaluation steps. Typically, however, the structure of the system is not known, or, even if it is known, the system is too complex to allow direct modelling. In this case, the black-box approach described in the next section is more appropriate.

5.4 Fitting Measurement Traces with PH Distributions

In the previous section we assumed that we know the internal structure of the system, and that it could be described based on the intuitive Dining Philosopher's problem. Typically, however, we will not know such details about the system under study, and will be limited to outside observations and measurements. With our example, we might not be able to observe why the system ran into deadlock, but we are certainly able to measure the length of the intervals between successive deadlocks. Similar situations often arise with measurements of delays, message lengths, or other phenomena, where we do not know the underlying causes, but can measure their effects. In such cases we can fit a phase-type distribution to the data in question, and use this model in our evaluation.

Consider Fig. 5.7, where we show both a histogram of some data and the density of a phase-type distribution approximating the data. Our aim is to approximate the data as closely as possible, in order to obtain correct results when using the approximating distribution later on. In this section we provide the basics for fitting data sets with phase-type distributions. We discuss costs, quality metrics, and introduce three established fitting tools.

5.4.1 Costs of Fitting PH Distributions to Data

Since a PH distribution is defined by the tuple $(\boldsymbol{\alpha}, \mathbf{A})$, the problem of fitting translates to finding an initial probability vector $\boldsymbol{\alpha}$ and a sub-generator matrix $\mathbf{A}$ of appropriate size n. While, in general, higher-order PH distributions can provide a better approximation [124], they are more expensive in both analysis and simulation. Furthermore, the time required for fitting a distribution increases with n, as more parameters have to be fitted. Consequently, careful choice of n is important.

As will be shown in Sects. 5.5 and 5.6, the cost of using a PH distribution depends not only on the size n, but also on the structure of the representation. The same holds for the fitting problem. Here, the number of free parameters to be fitted can be reduced significantly by choosing an appropriate representation: if we assume the size n of the representation to be constant, then general phase-type distributions in an arbitrary Markovian representation have $n + n^2$ free parameters, as $\boldsymbol{\alpha}$ is a row vector

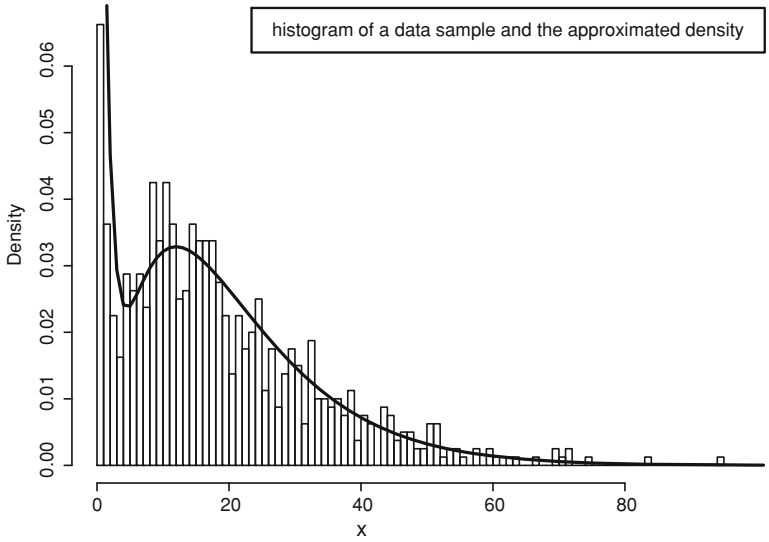

Fig. 5.7 Example data and its approximation with a phase-type distribution

of length n, and $\mathbf{A}$ is a matrix of size $n \times n$. If we assume that the representation is Monocyclic, we have a chain of m Feedback-Erlang blocks, each with a length parameter b_j, rate parameter λ_j and feedback probability z_j, and an initial probability vector of size n. As $m \leq n$, the upper limit for the number of free parameters is $3n+n$. Limiting ourselves to the APH class, we can utilise the CF-1 canonical form, which has only $2n$ free parameters: n transition rates and n initial probabilities. Finally, if we consider only HErD distributions in representations as shown in Fig. 5.3, the number of free parameters reduces to $3m$: m initial probabilities for the m Erlang branches, m lengths for the Erlang branches, and m transition rates.

5.4.2 Quality Measures

Fitting a phase-type distribution to data requires careful choice of the right fitting tool, as well as of fitting parameters such as sub-class and size. As just discussed, the approximation problem becomes less complex if data is fitted with subclasses of phase-type distributions, however, fitting quality may decrease as well, as subclasses cannot represent all properties of the general PH class. For example, hyperexponential distributions cannot approximate distributions with oscillating densities [880].

In order to assess the quality of data approximation, quality measures are required. An intuitive method consists in simply comparing the shape of the empirical PDF or CDF to that of the approximating PH distribution. This gives a visual impression how

Table 5.1 Performance measures defined in [572]

Performance measure	Definition
Area difference between distribution functions ΔF	$\Delta F = \int_0^\infty \lvert\hat{F}(x) - F(x)\rvert dt$
Area difference between densities Δf	$\Delta f = \int_0^\infty \lvert\hat{f}(x) - f(x)\rvert dt$
Relative error in the first moment (mean c_1)	$e_1 = \frac{\lvert\hat{c_1} - c_1\rvert}{c_1}$
Relative error in the second central moment (variance c_2)	$e_2 = \frac{\lvert\hat{c_2} - c_2\rvert}{c_1}$
Relative error in the third central moment (skewness c_3)	$e_3 = \frac{\lvert\hat{c_3} - c_3\rvert}{c_3}$

well the approximating PH distribution reflects the shape of the empirical PDF/CDF. For instance, in Fig. 5.7 the approximated density fits the data quite well.

While a visual impression often yields a good initial assessment, a more formal approach requires exact definitions of quality measures. In the area of PH fitting there exists a set of standard quality measures, as defined in [572]. These measures are summarised in Table 5.1: the first two performance measures formalise the visual comparison of empirical and approximated data, by computing the distance between both curves. The last three measures capture how well the fitted distribution approximates the empirical moments of the data. Based on these performance measures we can decide which tool to use, and which fitting is most appropriate for the requirements and future application of approximation results. For instance, for use in a stochastic model whose behaviour primarily depends on the first three moments, one would aim to get small relative moment errors, while in other applications fitting the shape of the density may be more important.

5.4.3 Introduction to PH-Fitting Tools

Here we outline three tools for data approximation with phase-type distributions: Moment Matching, G-FIT and PhFit. They mainly differ with respect to the algorithms they employ and the subclass of PH distributions they support. There are two general and relevant classes of algorithms: analytical and statistical methods, where the former relies on direct computation of the parameters and the latter is based on iterative procedures for parameter estimation.

5.4.3.1 Analytic Approximation: Moment Matching

Analytic moment-matching methods have the advantage of being fast, easy to implement, and giving low errors in the moments. On the other hand, accuracy of the fitting may be limited by the representation. We illustrate this using the method proposed in [870], which can fit an APH(2) distribution to the first three moments of a data set.

The approach proceeds by computing the approximation parameters directly from the moments, as follows: An APH(2) in CF-1 form with $\boldsymbol{\alpha} = (\alpha_1, 1-\alpha_1)$ and

$$\mathbf{A} = \begin{pmatrix} -\lambda_1 & \lambda_1 \\ 0 & -\lambda_2 \end{pmatrix},$$

is defined by three parameters, λ_1, λ_2, and α_1. Recall from Definition 5.4 the general moments-generating function for a PH distribution. Writing the first three moments explicitly:

$$\begin{aligned} E[X] = m_1 &= \frac{\lambda_1 + \alpha_1\lambda_2}{\lambda_1\lambda_2}, \\ E[X^2] = m_2 &= \frac{2(\lambda_1^2 + \alpha_1\lambda_1\lambda_2 + \alpha_1\lambda_2^2)}{\lambda_1^2\lambda_2^2}, \\ E[X^3] = m_3 &= \frac{6(\lambda_1^3 + \alpha_1\lambda_1^2\lambda_2 + \alpha_1\lambda_1\lambda_2^2 + \alpha_1\lambda_2^3)}{\lambda_1^3\lambda_2^3}, \end{aligned}$$

[870] obtain a system of three linear equations. Solving this system for $\lambda_1, \lambda_2, \alpha_1$ yields an APH(2) that matches the first three moments. However, possible solutions are limited by the moment bounds for the APH(2) class (cf. Sect. 5.2.3). For combinations of moments outside the moment bounds, the system has no solution, i.e., data sets with these moments cannot be fitted exactly by an APH(2). For instance, as follows from (5.5), the smallest SCV cv^2 that can be represented by an APH(2) is

$$cv^2 = \frac{1}{2},$$

which puts constraints on the relation of the mean and variance. Data sets with $cv^2 < \frac{1}{2}$ require PH distributions of higher order. Similar constraints exist for the third moment, although in some cases the third moment can be approximated even when no exact fitting is possible.

Iterative procedures for PH fitting have the advantage of providing more flexibility than analytical moment-matching methods. On the other hand, they are usually slower than the analytical approach. In the following we discuss two important tools of this class.

5.4.3.2 G-FIT for Fitting Hyper-Erlang Distributions

The G-FIT tool [880] approximates data using Hyper-Erlang distributions. Recall that the number of transition rates and the size of the initial vector of a Hyper-Erlang distribution only depend on the number of Erlang branches. This enables an efficient

fitting method: once the number m and length b of the Erlang branches have been set, the parameters are

$$\boldsymbol{\Theta} = (\boldsymbol{\beta}, \boldsymbol{\lambda}).$$

In each iteration the Expectation Maximisation (EM) algorithm (cf. [281]) computes parameters $\boldsymbol{\beta}$ and $\boldsymbol{\lambda}$ which maximise the likelihood of the parameters, given the data. G-FIT provides convergence checks based on the maximal change in $\boldsymbol{\Theta}$ and on the relative differences of the log-likelihood between successive iterations.

The user may specify the number and length of Erlang branches prior to fitting or let G-FIT determine an optimal size. In the first case the user has to set a number of Erlang branches and their length. The second option is more general and is useful for the unexperienced user. It requires as input only a number of phases for the resulting distribution. G-FIT will then estimate optimal number of Erlang branches and their parameters, by trying all possible combinations.

G-FIT expects an input as a text file containing the data set. The first line should be a number of data points in the data set followed by data points themselves, which are given one per line. The output is also a text file, containing the number of Erlang branches, number of phases, initial probabilities and transition rates for each Erlang branch.

5.4.3.3 PhFit for Fitting APH Distributions

The PhFit tool [446] approximates data using acyclic phase-type distributions in CF-1 form. It applies a variant of the Frank/Wolfe algorithm [355, 361] for constrained non-linear minimisation of the distance between the PH distribution and the data. One major advantage is that the user can choose between different distance measures, in order to obtain an optimal fitting. The distance measures supported by PhFit are the relative entropy, PDF area distance, and CDF area distance, defined as

$$\int_0^\infty f(t)\log(\frac{f(t)}{\hat{f}(t)})dt, \quad \int_0^\infty |\hat{F}(x) - F(x)|dt, \quad \text{and} \quad \int_0^\infty |\hat{f}(x) - f(x)|dt, \text{ respectively,}$$

where $f(t)$ denotes the probability density function (PDF) of the original distribution and $\hat{f}(t)$ the PDF of the approximating distribution, $F(t)$ the cummulative distribution function (CDF) of the original distribution and $\hat{F}(t)$ the CDF of the fitted distribution. Among the fitting tools we discuss, PhFit is the only one with a graphical user interface. This feature is beneficial for finding appropriate fitting parameters and evaluation of results.

PhFit computes optimal values for the distribution parameter $(\boldsymbol{\alpha}, \mathbf{A})$ starting with special initial values $(\boldsymbol{\alpha}^{(0)}, \mathbf{A}^{(0)})$ according to the distance measure. PhFit picks optimal values from 1,000 randomly generated pairs of vectors. The distance measure

defines the optimality criterion. The optimisation problem is solved by using the iterative linearisation method. After linearisation in a local neighbourhood of the current distribution parameters, the direction for optimisation of the distance measure is determined by the Simplex algorithm. The algorithm stops computation once the relative difference between

$$(\boldsymbol{\alpha}^{(i-1)}, \mathbf{A}^{(i-1)})$$

and

$$(\boldsymbol{\alpha}^{(i)}, \mathbf{A}^{(i)})$$

for iteration i is less than the predefined value, or if a maximum number of iterations is reached.

PhFit provides separate fitting for body and tail. The body is the part of distribution with the most mass, whereas the tail represents rare data points. The user can choose the boundary where the tail begins. The tail is approximated with a heuristic method that determines parameters for a hyper-exponential distribution. The body is then approximated as described before. The resulting distribution is then given by the CF-1 form and the hyper-exponential distribution.

PhFit requires as input a text file containing the data in ascending order. The output consists of the initial probability vector $\boldsymbol{\alpha}$ and the diagonal of the subgenerator matrix. Note, however, that in contrast to the definition we gave in Definition 5.3, PhFit considers the 0th state to be absorbing, instead of state $(n + 1)$. That is, the output of PhFit is reversed, compared to the notation used throughout this chapter.

5.5 Phase-Type Distributions in Model Analysis: Matrix Analytic Methods

In the previous sections we discussed how a phase-type distribution modelling the phenomenon of interest can be obtained either explicitly or by fitting measurement traces. We will now illustrate how such a model can be used in analytical approaches. Referring to our example with the faulty server, we may want to analyse the effect of deadlocks on job processing in a queueing system. We assume that the server is reset after a deadlock, but that the fault leading to the deadlock persists. Then, the instances of deadlock can be fitted by a PH renewal process or a Markovian Arrival Process (MAP). If the process starts from the same initial state each time the process of deadlocks will be uncorrelated and forms a PH renewal process. In case of correlated initial states the time between the deadlocks forms a MAP. In this section we discuss matrix-analytic methods [579] for analysing complex models using phase-type distributions.

Matrix-analytic methods utilize the structure of the Markov chain which, in this chapter, is two-dimensional. Both dimensions have their own characteristics. The first dimension represents the—usually finite—number of phases $J(t)$ of the process. The second dimension, denoted by $N(t)$, is the infinite counting process. This approach results in an infinite, but well-structured, Markov chain on the block level where the blocks describe the phase either with or without arrival. The same block structure appears also in the generator matrix of the Markov chain which can be upper block-bidiagonal or tridiagonal in our cases.

The examples of this section show how the matrix-analytic methods utilize the analytic PH properties during the solution of complex Markov models. The result can be either the short-term or the steady-state behavior. The methods also allow to find the solution of infinite models by solving finite problems.

5.5.1 Processes with PH Marginal Distribution

A sequence of random variables—according to a given (marginal) distribution—defines a stochastic process or simply process. Processes play an important role in stochastic modeling thus it comes naturally to propose the process with PH marginal distribution. Here we investigate both the independent identical distributed (iid) and the correlated arrival process with PH marginal distribution. These are the PH renewal process and the Markov arrival process (MAP) respectively.

Referring to the example in Sect. 5.3 a faulty server has PH distributed time to deadlock if the relevant times are exponentially distributed. Furthermore if the system restarts at deadlock situations then the resulting sequence of times to deadlock is a stochastic process with PH marginal distribution.

5.5.1.1 PH Renewal Process

Given a phase-type distribution represented by the initial vector $\boldsymbol{\alpha}$ and subgenerator matrix $\mathbf{A}$, the generator matrix

$$\mathbf{Q} = \begin{pmatrix} \mathbf{A} & \mathbf{a}\boldsymbol{\alpha} & 0 & \dots & \\ 0 & \mathbf{A} & \mathbf{a}\boldsymbol{\alpha} & 0 & . \\ 0 & 0 & \mathbf{A} & \mathbf{a}\boldsymbol{\alpha} & 0 \\ \dots & \dots & \dots & \dots & \dots \end{pmatrix}, \tag{5.6}$$

defines the PH renewal process for which $(\boldsymbol{\alpha}, \mathbf{A})$ is the marginal distribution. $\mathbf{a} = -\mathbf{A}\mathbb{1}$ is the vector of absorption rates of the marginal distribution. The blocks on the diagonal describe the phase transitions of the PH marginal, and the blocks in the upper co-diagonal describes the phase transitions belonging to the renewal instances. The graph of the corresponding continuous time Markov chain (CTMC) is depicted in Fig. 5.8.

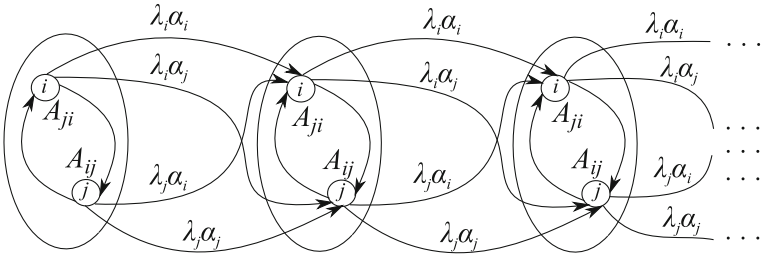

Fig. 5.8 The graph of the PH renewal process

The product in the upper co-diagonal blocks expresses that the initial distribution of the next interarrival is always the same ($\boldsymbol{\alpha}$) after arrival ("absorption" in the PH marginal) regardless of any of the other interarrivals, i.e., the process is uncorrelated.

The generator matrix of the phase process is $\mathbf{H} = \mathbf{A} + \mathbf{a}\boldsymbol{\alpha}$. The steady state phase distribution ($\boldsymbol{\pi}$) is the solution of the linear system of equations

$$\begin{aligned} \boldsymbol{\pi}\mathbf{H} &= 0 \\ \boldsymbol{\pi}\mathbb{1} &= 1. \end{aligned} \tag{5.7}$$

The transient phase distribution is

$$\boldsymbol{\pi}(t) = \boldsymbol{\pi}(0)\mathrm{e}^{\mathbf{H}t} \tag{5.8}$$

which is a vector of elements $\pi_i(t) = \Pr(J(t) = i)$ giving the probability that the process is in phase i at time t. Using the transient phase behavior, at time t the remaining time to the next arrival is distributed according to the phase-type distribution $(\boldsymbol{\pi}(t), \mathbf{A})$.

Let the entries of the vector $\boldsymbol{\pi}(n, t) = (\Pr(N(t) = n, J(t) = j))$ give the probabilities that at time t the number of arrivals is equal to n and the level process is in phase j. With initial conditions $\boldsymbol{\pi}(0, 0) = \boldsymbol{\alpha}$ and $\boldsymbol{\pi}(i, 0) = 0$ $(i > 0)$, the transient number of arrivals is given by the differential equation

$$\frac{d\boldsymbol{\pi}(i, t)}{dt} = \boldsymbol{\pi}(i, t)\mathbf{A} + \boldsymbol{\pi}(i - 1, t)\mathbf{a}\boldsymbol{\alpha}, \tag{5.9}$$

whose z-transform, with initial condition $\boldsymbol{\pi}(z, 0) = \boldsymbol{\alpha}$, is

$$\frac{d\boldsymbol{\pi}(z, t)}{dt} = \boldsymbol{\pi}(z, t)\mathbf{A} + z\boldsymbol{\pi}(z, t)\mathbf{a}\boldsymbol{\alpha} = \boldsymbol{\pi}(z, t)\left(\mathbf{A} + z\mathbf{a}\boldsymbol{\alpha}\right). \tag{5.10}$$

The solution of the differential equation, i.e., the transient distribution of the number of arrivals, is

$$\boldsymbol{\pi}(z, t) = \boldsymbol{\alpha}\mathrm{e}^{(\mathbf{A}+z\mathbf{a}\boldsymbol{\alpha})t}. \tag{5.11}$$

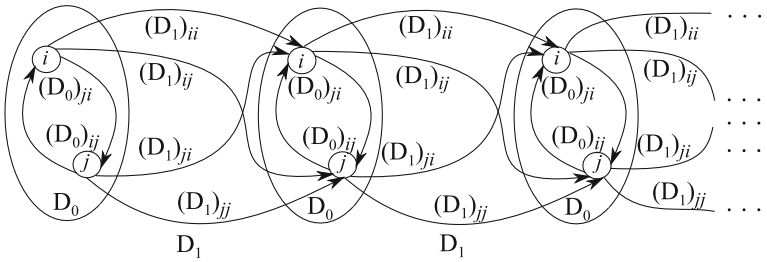

Fig. 5.9 The graph of the Markov arrival process

Regarding to the example in Sect. 5.3, the results on the PH renewal processes can be used to model the faulty server if the system restarts in the same (initial) state after each deadlock situation.

5.5.1.2 Markov Arrival Process

In the PH renewal process the phase distribution is the same after every arrival. In contrast, in the Markov Arrival Process (MAP) after each arrival an arbitrary phase distribution may hold. This allows the modelling of correlated arrival processes. The two-dimensional CTMC of the MAP process is also defined by the phase process $J(t)$, describing the phase of the marginal distribution, and by the counting process $N(t)$, giving the number of arrivals. Its graph is depicted in Fig. 5.9 and its generator matrix is

$$\mathbf{Q} = \begin{pmatrix} \mathbf{D}_0 & \mathbf{D}_1 & 0 & \dots & \dots \\ 0 & \mathbf{D}_0 & \mathbf{D}_1 & 0 & . \\ 0 & 0 & \mathbf{D}_0 & \mathbf{D}_1 & 0 \\ \dots & \dots & \dots & \dots & \dots \end{pmatrix}, \tag{5.12}$$

where the Markov arrival process is represented by $\mathbf{D}_0$—the phase transitions without arrival—and $\mathbf{D}_1$—the phase transitions with one arrival. Such a MAP is denoted as MAP $(\mathbf{D}_0, \mathbf{D}_1)$.

The interarrival times of the MAP $(\mathbf{D}_0, \mathbf{D}_1)$ are PH $(\alpha_0, \mathbf{D}_0)$, PH $(\alpha_1, \mathbf{D}_0) \dots$ The—correlated—phase distribution embedded at arrival instances forms a discrete time Markov chain (DTMC) with state transition probability matrix $\mathbf{P} = (-\mathbf{D}_0)^{-1}\,\mathbf{D}_1$.

The joint probability density function of the interarrival times, X_0 and X_k, is

$$f_{X_0,X_k}(x_0, x_k) = \boldsymbol{\pi} \mathrm{e}^{\mathbf{D}_0 x_0} \mathbf{D}_1 \mathbf{P}^{k-1} \mathrm{e}^{\mathbf{D}_0 x_k} \mathbf{D}_1 \mathbb{1}, \tag{5.13}$$

where $\boldsymbol{\pi}$ is the embedded stationary phase distribution at arrival instances, i.e., it is the solution of the linear system of equations

$$\begin{aligned} \boldsymbol{\pi}\mathbf{P} &= \boldsymbol{\pi} \\ \boldsymbol{\pi}\,\mathbb{1} &= 1. \end{aligned} \tag{5.14}$$

The stationary interarrival time distribution is PH $(\boldsymbol{\pi}, \mathbf{D}_0)$ with nth moment

$$E\left[X^n\right] = n!\boldsymbol{\pi}\,(-\mathbf{D}_0)^{-n}\,\mathbb{1} \tag{5.15}$$

and the joint moment of two interarrivals is

$$\begin{aligned} E\left[X_0 X_k\right] &= \int_{x_0}\int_{x_k} x_0 x_k \boldsymbol{\pi} \mathrm{e}^{\mathbf{D}_0 x_0}\mathbf{D}_1\mathbf{P}^{k-1}\mathrm{e}^{\mathbf{D}_0 x_k}\mathbf{D}_1\,\mathbb{1}dx_0 dx_k \\ &= \boldsymbol{\pi}\,(\mathbf{D}_0)^{-1}\,\mathbf{P}^k\,(\mathbf{D}_0)^{-1}\,\mathbb{1}. \end{aligned} \tag{5.16}$$

The covariance of two interarrivals is

$$\mathrm{cov}\,(X_0, X_k) = \mathrm{E}\left[X_0 X_k\right] - \mathrm{E}^2\left[X\right] \tag{5.17}$$

and using (5.15)–(5.17) the lag k correlation of the MAP is

$$\mathrm{corr}\,(X_0, X_k) = \frac{\mathrm{cov}\,(X_0, X_k)}{\mathrm{E}\left[X^2\right] - \mathrm{E}^2\left[X\right]}. \tag{5.18}$$

The MAP can help to model the faulty server of Sect. 5.3 if the initial states of the system (after restart) are correlated.

5.5.2 The Quasi Birth-Death Process

The quasi birth-death (QBD) process [579, 683] is also defined by the phase process $(J(t))$ and the counting process $(N(t))$. But in case of the QBD process the counting, or the "level", process is allowed to be decreased by one as well as to stay on the same level or to be increased by one. It is thus the "multiphase" extension of the birth-death process which is for example the solution of the M/M/1 queueing system. The generator matrix of the QBD process has block-tridiagonal form

$$\mathbf{Q} = \begin{pmatrix} \mathbf{L}' & \mathbf{F} & 0 & .. & . \\ \mathbf{B} & \mathbf{L} & \mathbf{F} & 0 & . \\ 0 & \mathbf{B} & \mathbf{L} & \mathbf{F} & 0 \\ \cdots & \cdots & \cdots & \cdots & .. \end{pmatrix}, \tag{5.19}$$

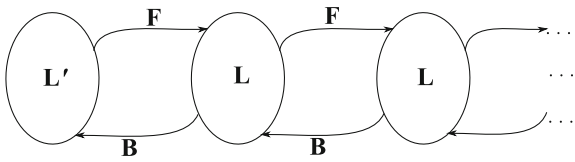

Fig. 5.10 The graph of the quasi birth-death process

where the blocks or level transition matrices are

- $\mathbf{L}'$ local state transitions inside the first—irregular—block,
- $\mathbf{B}$ backward (level) state transitions,
- $\mathbf{L}$ local state transitions on the regular levels and
- $\mathbf{F}$ forward (level) state transitions.

The graph of the QBD is depicted in Fig. 5.10.

We give the solution method of the QBD through the analysis of the MAP/PH/1 queueing system with arrival process MAP $(\mathbf{D}_0, \mathbf{D}_1)$ and service time PH $(\boldsymbol{\alpha}, \mathbf{A})$. The level transition matrices are

$$\begin{aligned} \mathbf{L}' &= \mathbf{D}_0 \otimes \mathbf{I} \\ \mathbf{B} &= \mathbf{I} \otimes \mathbf{a}\boldsymbol{\alpha} \\ \mathbf{L} &= \mathbf{D}_0 \oplus \mathbf{A} \\ \mathbf{F} &= \mathbf{D}_1 \otimes \mathbf{I}, \end{aligned}$$

where $\mathbf{a} = -\mathbf{A}\mathbb{1}$ and $\mathbf{I}$ is the identity matrix of appropriate size. The operators $\otimes$ and $\oplus$ are the Kronecker product and sum, respectively.

The generator matrix of the phase process is $\mathbf{H} = \mathbf{B}+\mathbf{L}+\mathbf{F}$ and if it is irreducible then the steady state phase distribution is the solution of the linear system of equations

$$\begin{aligned} \boldsymbol{\pi}\mathbf{H} &= 0 \\ \boldsymbol{\pi}\mathbb{1} &= 1. \end{aligned} \tag{5.20}$$

The QBD process is stable if its stationary drift is less than zero

$$d = \boldsymbol{\pi}\mathbf{F}\mathbb{1} - \boldsymbol{\pi}\mathbf{B}\mathbb{1} < 0. \tag{5.21}$$

The steady state solution of the QBD is the solution of the infinite system of linear equations

$$\begin{aligned} \boldsymbol{\nu}\mathbf{Q} &= 0 \\ \boldsymbol{\nu}\mathbb{1} &= 1. \end{aligned} \tag{5.22}$$

Partitioning $\boldsymbol{\nu}$ according to the blocks of $\mathbf{Q}$ is

$$\boldsymbol{\nu} = \left(\boldsymbol{\nu}_0\ \boldsymbol{\nu}_1\ \boldsymbol{\nu}_2\ \ldots\right)$$

and substituting the partitions into (5.22) we get

$$\boldsymbol{\nu}_0\mathbf{L}' + \boldsymbol{\nu}_1\mathbf{B} = 0 \tag{5.23}$$

and

$$\boldsymbol{\nu}_{i-1}\mathbf{F} + \boldsymbol{\nu}_i\mathbf{L} + \boldsymbol{\nu}_{i+1}\mathbf{B} = \mathbf{0} \quad \forall \mathbf{i} \geq \mathbf{1}. \tag{5.24}$$

Assuming that the Markov chain is irreducible $\nu_i = \nu_{i-1}\mathbf{R} = \nu_0\mathbf{R}^i \quad (\forall i)$, i.e., its solution is the matrix geometric distribution, the general Eq. (5.24) can be rewritten as

$$\begin{aligned}
&\boldsymbol{\nu}_0\mathbf{R}^{i-1}\mathbf{F} + \boldsymbol{\nu}_0\mathbf{R}^i\mathbf{L} + \boldsymbol{\nu}_0\mathbf{R}^{i+1}\mathbf{B} = 0 \\
&\boldsymbol{\nu}_0\mathbf{R}^{i-1}\left(\mathbf{F} + \mathbf{RL} + \mathbf{R}^2\mathbf{B}\right) = 0
\end{aligned}$$

with a solution determined by

$$\mathbf{F} + \mathbf{RL} + \mathbf{R}^2\mathbf{B} = 0. \tag{5.25}$$

If the QBD is stable there is one of the solutions of $\mathbf{R}$ whose eigenvalues are within the unit circle on the complex plane.

As all the eigenvalues of the relevant $\mathbf{R}$ are within the unit circle there exists the limit of the sum $\sum_{i=0}^{\infty}\mathbf{R}^i = (\mathbf{I} - \mathbf{R})^{-1}$. Using the convergence the normalizing condition of $\boldsymbol{\nu}$ can be expressed as

$$\boldsymbol{\nu}\,\mathbb{1} = \sum_{i=0}^{\infty}\boldsymbol{\nu}_i\,\mathbb{1} = \sum_{i=1}^{\infty}\boldsymbol{\nu}_0\mathbf{R}^i\,\mathbb{1} = \boldsymbol{\nu}_0\sum_{i=1}^{\infty}\mathbf{R}^i\,\mathbb{1} = \boldsymbol{\nu}_0\,(\mathbf{I} - \mathbf{R})^{-1}\,\mathbb{1} = 1. \tag{5.26}$$

Now substituting $\mathbf{R}$ into (5.23) and using (5.26) we have a linear system of equations

$$\begin{aligned}
&\boldsymbol{\nu}_0\left(\mathbf{L}' + \mathbf{RB}\right) = 0 \\
&\boldsymbol{\nu}_0\,(\mathbf{I} - \mathbf{R})^{-1}\,\mathbb{1} = 1
\end{aligned} \tag{5.27}$$

for the zeroth block of ν. All the other blocks can be calculated using ν_0 and $\mathbf{R}$ as

$$\boldsymbol{\nu}_i = \boldsymbol{\nu}_0\mathbf{R}^i, \quad \forall i. \tag{5.28}$$

By these considerations the infinite problem of solving the QBD in (5.22) is reduced to be the solution of the finite problems in (5.25), (5.27) and (5.28).

5.6 Phase-Type Distributions in Random-Variate Generation

While phase-type distributions enable efficient solutions for analytical models, they have applications beyond analytical approaches. Their ability to provide good models for many different empirical distributions makes them attractive in evaluation techniques where observed phenomena must be represented accurately and efficiently. In particular, they can be used both in discrete-event simulation of models that cannot be solved by analytical methods, and in fault-injection-driven experiments in testbeds. Referring back to our example, in addition to considering an analytical solution we might want to explore the effect of the faulty server on the resilience of our system by running a simulation or performing measurements in a testbed. Then, we need to generate random variates from a PH distribution describing the times to deadlock.

Phase-type distributed samples may be generated by playing the CTMC until absorption, and by numerical inversion of the distribution function [157]. In the following we focus on methods that 'play' the CTMC. Note that these methods require the Markovian representation.

The methods discussed in the following utilise random variates from the uniform, exponential, Erlang, and geometric distributions. We assume that random variates with uniform distribution on $(0, 1)$ are given, and denote these by U. Using the inversion method, a sample with exponential distribution with rate λ is then drawn by

$$\mathrm{Exp}(\lambda) = -\frac{1}{\lambda}\ln(U).$$

A sample from the Erlang distribution with degree b and rate λ is generated by

$$\mathrm{Erl}(b, \lambda) = -\frac{1}{\lambda}\ln\left(\prod_{i=1}^{b} U_i\right).$$

Note that this way of sampling $\mathrm{Erl}(b, \lambda)$ is more efficient than the functional equivalent of drawing b exponentially distributed samples and summing them up, because the ln operation is applied only once. Finally, a sample from the geometric distribution (starting from 0) with parameter p is obtained by

$$\mathrm{Geo}(p) = \left\lfloor \frac{\ln(U)}{\ln(p)} \right\rfloor.$$

The most natural way to generate a PH-distributed sample by playing the CTMC proceeds as follows: first, we select a state i by drawing an integer sample distributed

according to the initial probability vector $\boldsymbol{\alpha}$. Afterwards, in each step the next state is selected according to the next-state probability vector, which is given by the ith row of the embedded Markov chain of $\overline{\mathbf{A}}$,

$$\mathbf{S} = \mathbf{I} - diag(\overline{\mathbf{A}})^{-1}\overline{\mathbf{A}}.$$

In the following, let $\mathbf{S}_i$ denote the ith row vector of $\mathbf{S}$. The sojourn time for state i is obtained as a sample from the exponential distribution with rate $-\lambda_{ii}$. Letting $\boldsymbol{e}_i$ denote the row vector with 1 at position i, and 0 everywhere else, the `Play` method can be given in pseudocode as follows:

`Procedure Play:`

1) $x := 0$. Draw an $\boldsymbol{\alpha}$-distributed discrete sample i for the initial state.
2) The chain is in state i
   - draw an $\mathbf{S}_i$-distributed discrete sample for the next state,
   - $x+ = \text{Exp}(-\lambda_{ii})$,
   - if the next state is the absorbing one ($i = n + 1$) go to 3), otherwise go to 2)
3) Return x.

In [684], Neuts and Pagano point out that when traversing a state more than once, the `Play` method adds up multiple samples from the same exponential distribution. The sum of k_i exponential distributions of the same rate $-\lambda_{ii}$, however, is the Erlang distribution with length k_i and rate $-\lambda_{ii}$. As shown above, drawing a sample from the Erlang distribution of length k_i requires only one logarithm operation, as opposed to k_i logarithms when drawing individual exponential samples. Thus, Neuts and Pagano propose the following method, which, instead of drawing exponential samples for each visit to a state i, counts the number of visits and then draws one Erlang-distributed sample for each state:

`Procedure Count:`

1) $x := 0$, $k_i := 0$, $(i = 1, \ldots, n)$, Draw an $\boldsymbol{\alpha}$-distributed discrete sample i for the initial state.
2) The chain is in state i
   - k_i += 1,
   - draw an $\mathbf{S}_i$-distributed discrete sample for the next state,
   - if the next state is the absorbing one go to 3) otherwise to 2)
3) for $i = 1, \ldots, n$; do x += Erl(k_i, $-\lambda_{ii}$); done
4) Return x.

If the distribution is in Monocyclic form, we can derive another method from the structural properties of the Monocyclic representation. Recall that this representation consists of a chain of Feedback-Erlang blocks. With such a chain, possible state transitions are predetermined by the structure in two ways: First, when we leave a Feedback-Erlang block j, the next state will be the first state of the next Feedback-Erlang block $j+1$. This implies that no new sample is required for choosing the successor block. Second, recall from Fig. 5.5 that each FE block consists of a chain of $m_j - 1$ states with exactly one outgoing transition (to the next state), and only one state with two outgoing transitions (the feedback state). Thus, within each FE block the only state where the next state is not determined by the structure is the last one. Furthermore, as the last state has only two outgoing transitions, the choice of staying within block j or entering the next block $j+1$ corresponds to a Bernoulli experiment with parameter z_j. Consequently, the number of 'loops' in each block follows a geometric distribution with parameter z_j. Therefore, in order to generate the sample corresponding to the jth Feedback-Erlang block, we add a geometrically distributed number of exponentially distributed random variates with the same rate λ_j. As discussed when introducing the `Count` method, an efficient way of doing this is to draw a sample from an Erlang distribution of the appropriate length. These considerations lead to the following method:

`Procedure Monocyclic:`

1) $x := 0$. Draw an $\boldsymbol{\alpha}$-distributed discrete sample for the initial state,
2) the chain is in state l of block i (for the left-most state of the block, $l = b_i$)
   - $c = \text{Geo}(z_i)$,
   - $x+ = \text{Erl}(cb_i + l, \lambda_i)$
   - if the next block is the absorbing state go to 3), otherwise $l = b_{i+1}$, $i = i + 1$ and go to 2)
3) Return x.

The first three methods are applicable to general PH distributions. If we restrict our attention to sub-classes, more efficient methods can be designed. First, consider the APH class in CF-1 form. As a special case of the Monocyclic form, the CF-1 form is a chain of states, where each state has exactly one successor state (cf. Fig. 5.4), and thus the next state is not chosen randomly. Hence, once an initial state has been selected, the random variate is simply the sum of exponentially distributed samples from each of the successor states[3]:

[3] Note that the transition rates in the CF-1 form are usually not identical, hence we cannot simply draw an Erlang-distributed sample.

`Procedure Simpleplay:`

1) $x := 0$. Draw an $\boldsymbol{\alpha}$-distributed discrete sample for the initial state.
2) The chain is in state i.
   - $x+ = \text{Exp}(-\lambda_{ii})$,
   - $i+ = 1$,
   - if the next state is the absorbing state go to 3), otherwise go to 2).
3) Return x.

If we assume a Hyper-Erlang distribution, represented as shown in Fig. 5.3, we can simplify the procedure `Count`, by using our knowledge that each of the branches is an Erlang distribution:

`Procedure SimpleCount:`

1) Draw a $\boldsymbol{\beta}$-distributed discrete sample to choose an Erlang branch i.
2) Return $\text{Erl}(b_i, \lambda_i)$.

5.6.1 Costs of Generating PH-Distributed Numbers

In the previous section we argue that the methods for generating random variates differ in their efficiency. We will now treat the costs of random number generation from phase-type distributions in a more formal way. All of the algorithms use exponential random variates for the sojourn times and uniform random variates for choosing the initial state. `Play` and `Count` additionally use uniform random variates for choosing successor states, while the `Monocyclic` algorithm needs geometrically distributed numbers for the number of loops in each Feedback-Erlang block. In order to draw from an exponential or geometric distribution, we need uniform random variates and logarithm operations. Therefore, we define the following two metrics for measuring algorithm complexity:

Definition 5.6 Let $\#uni$ be the number of uniform random variates that need to be generated and let # ln be the number of logarithm operations that must be performed for generating one PH-distributed random variate from a given PH distribution $(\boldsymbol{\alpha}, \mathbf{A})$.

Using these metrics, we can compare the complexity of the algorithms. We consider both worst-case and average costs.

Table 5.2 Theoretical costs of generating PH distributed random variates from different PH classes and using different PH representations (where $\boldsymbol{\nu} = (n, n-1, \ldots, 1)$, $n^* = \boldsymbol{\alpha}(\text{diag}(\mathbf{A})^{-1}\mathbf{QB})^{-1}\mathbb{1}$)

PH Class	Worst case		Average case	
	#uni	#ln	#uni	#ln
HEx(n) `SimpleCount`	2	1	2	1
HErD(n) `SimpleCount`	$\max\{b_i + 1\}$	1	$\boldsymbol{\beta}\mathbf{b}^\mathsf{T} + 1$	1
APH(n) `SimplePlay`	$n + 1$	n	$\boldsymbol{\alpha}\boldsymbol{\nu}^\mathsf{T} + 1$	$\boldsymbol{\alpha}\boldsymbol{\nu}^\mathsf{T}$
PH(n) `Play`	∞	∞	$2\tilde{n} + 1$	$\tilde{n}$
PH(n) `Count`	∞	n	$2\tilde{n} + 1$	n
`Monocyclic`	∞	$3m$	$\boldsymbol{\omega}\boldsymbol{\varphi}^\mathsf{T} + \boldsymbol{\alpha}\boldsymbol{\psi}^\mathsf{T}$	$\boldsymbol{\omega}\boldsymbol{\vartheta}^\mathsf{T}$

5.6.1.1 Worst-Case Costs

Let $\tilde{n}$ denote the length of the longest possible path through the CTMC. For the `Play` method, we draw one exponentially distributed random variate for each traversed state, and hence need one logarithm and one uniform random variate per step, as well as an additional uniform for choosing the next state. For this method, $\#uni$ and $\#ln$ are proportional to $\tilde{n}$. However, $\tilde{n}$ is not defined if there are cycles in the CTMC. Therefore, worst-case costs are not defined for `Play`.

The same problem with the unknown maximum number of state traversals occurs with the `Count` method. However, in this case we only draw Erlang-distributed samples (one for each state). Therefore, the maximum number of logarithm operations is bounded by the number of states: $\#ln = n$. Similarly, for the `Monocyclic` method we draw one Erlang-distributed and one geometrically-distributed sample for each Feedback-Erlang block. The latter requires another two logarithm operations, in addition to the one for generating the Erlang sample. As the worst case occurs when we start in the first block, the worst-case number of traversed FE blocks is m, and thus $\#ln = 3m$.

For APH in CF-1 form and using the `SimplePlay` method, the worst case is if the chain is entered at state $i = 1$, since in that case we have to traverse the whole chain. Thus, $\tilde{n} = n$. Obviously, for a Hyper-Erlang distribution in CF-1 form, $\tilde{n} = n$ holds as well. However, if we consider the Hyper-Erlang form and simulation using the `SimpleCount` method, the worst case is equivalent to choosing the longest Erlang branch. In that case, $\tilde{n} = \max\{b_i\} \leq n$. The worst-case costs can be computed as follows: With every class, we need one uniform random variate to choose the initial state. When using the APH(n) class in CF-1 form we need $\tilde{n} = n$ uniforms and $\tilde{n} = n$ logarithms for the consecutive phases. With the HErD class and the `SimpleCount` method we need $\tilde{n} = \max\{b_i\}$ additional random variates and one logarithm to obtain an Erlang-distributed random number. We summarise these results in the left half of Table 5.2.

5.6.1.2 Average Costs

In general, we do not expect to have worst-case behaviour, but are more interested in average costs. This measure is based on the average number of state transitions up to absorption,

$$\bar{n} = \boldsymbol{\alpha}(\mathrm{diag}(\mathbf{A})^{-1}\mathbf{A})^{-1}\mathbb{1}.$$

Applying the `Play` method for the general PH class, in each step we need two uniform random variates (one for the exponential sample and one for choosing the next state, see above), and one logarithm operation. As before, applying the `Count` procedure instead, the number of logarithms is $\#ln = n$, while the number of uniforms stays $\#uni = \bar{n}$.

Canonical forms enable explicit expressions for $\bar{n}$. For Mono($\boldsymbol{\alpha}, m, \mathbf{b}, \boldsymbol{\lambda}, \mathbf{z}$) we introduce vector $\boldsymbol{\omega}$ of size m, whose ith element is the probability of starting from Feedback-Erlang block i (e.g., $\omega_1 = \sum_{j=1}^{b_1} \alpha_j$), vector $\boldsymbol{\varphi}$ of size m, whose ith element is $\varphi_i = \frac{z_i b_i}{1-z_i} + \sum_{j=i+1}^{m} \frac{b_j}{1-z_j}$ (the mean number of steps spent in a Feedback-Erlang block from the first feedback, i.e., excluding the steps from the initial state to the feedback state in the first passage through the initial block), vector $\boldsymbol{\psi}$ of size n whose ith element indicates how many phases are needed to reach the next Feedback-Erlang block (e.g., if $b_1 \geq 2$ then $\psi_1 = b_1$, $\psi_2 = b_1 - 1$).

Using these notations the mean number of steps till absorption is

$$\bar{n} = \boldsymbol{\omega}\boldsymbol{\varphi}^{\mathsf{T}} + \boldsymbol{\alpha}\boldsymbol{\psi}^{\mathsf{T}},$$

where $\boldsymbol{\alpha}\boldsymbol{\psi}^{\mathsf{T}}$ contains the number of steps if there is no feedback (i.e., if $z_i = 0$, for $i = 1, \ldots, m$) and $\boldsymbol{\omega}\boldsymbol{\varphi}^{\mathsf{T}}$ contains the additional number of steps due to the loops in the Feedback-Erlang block.

The mean number of ln operations is

$$\ell^* = \boldsymbol{\omega}\boldsymbol{\vartheta}^{\mathsf{T}},$$

where $\boldsymbol{\vartheta}$ is a row vector of size m whose ith element indicates the number of required ln operations starting from block i. $\vartheta_i = \sum_{j=i}^{m}(1 + 2\,\mathrm{sgn}(z_j))$, since a degenerate Feedback-Erlang block with $z_i = 0$ is Erlang(l, λ_i) distributed which requires one ln operation and a non degenerate ($z_i > 0$) Feedback-Erlang block requires three ln operations, two ln operations for $c = \mathrm{Geo}(z_i)$ and one for $\mathrm{Erl}(cb_i + l, \lambda_i)$.

For the APH class in CF-1 form, there exists an even simpler expression, as the number of traversed states depends only on the initial state, which in turn is determined by the initial probability vector $\boldsymbol{\alpha}$. Thus, for APH in CF-1 form,

$$\bar{n} = \boldsymbol{\alpha}\boldsymbol{\nu}^{\mathsf{T}}, \quad \text{where} \quad \boldsymbol{\nu} = (n, n-1, \ldots, 1).$$

Equivalently, for the HErD class, $\bar{n}$ is a weighted sum of the lengths of the Erlang branches:

$$\bar{n} = \boldsymbol{\alpha}\mathbf{b}^{\mathsf{T}}.$$

5.6.2 Optimisation

Considering the costs for the different methods discussed in the previous sections, it becomes clear that both the representation of the distribution and the method have an impact on the efficiency of PH random variate generation. One immediate question is then: what is the optimal representation to generate random variates efficiently? While the answer to this question is not yet available for the general PH case, [764] presents the following result for APH in CF-1 form:

Lemma 1 *[764] Given a Markovian representation* $(\boldsymbol{\alpha}, \boldsymbol{\Lambda})$ *in CF-1 form, the representation* $(\boldsymbol{\alpha}^*, \boldsymbol{\Lambda}^*)$ *that reverses the order of the rates is optimal with respect to* $\bar{n}$ *if* $\boldsymbol{\alpha}^*$ *is a stochastic vector. In this case, all bi-diagonal representations are Markovian.*

The proof given in [764] relies on the observation that swapping two adjacent rates λ_i, λ_{i+1} moves probability mass towards the end of the chain only if $\lambda_i < \lambda_{i+1}$. Thus, reversing the CF-1 order (where $\lambda_i \leq \lambda_{i+1}$ for all i) gives an initial probability vector $\boldsymbol{\alpha}$ where probability mass is concentrated at the higher indices. Recalling from above, $\bar{n} = \boldsymbol{\alpha}\boldsymbol{\nu}^{\mathsf{T}}$ for APH, i.e., high probability for states close to absorption implies low average costs.

Note, however, that reversing the CF-1 form may result in $\boldsymbol{\alpha}$ with negative entries [764]. In this case, the tuple $(\boldsymbol{\alpha}^*, \mathbf{A}^*)$ still represents the same distribution, but the representation does not have a Markovian interpretation anymore, and thus $\bar{n}$ is not defined, nor can `SimplePlay` be applied. The optimal ordering can then be found by exhaustive search over all $n!$ possible orderings, or by heuristics that try to find a Markovian representation that is as similar as possible to the reversed CF-1. The heuristics presented in [764] either start from the CF-1 form and apply pair-wise swappings until the result would be non-Markovian, or start from the reversed CF-1 and try to reach a Markovian representation.

5.7 Conclusion

In this chapter we introduced the basics of using phase-type distributions as tools in resilience evaluation, discussing the complete workflow from explicit derivation and fitting to application in both analytical and simulation methods. One application of the methods described here is illustrated in Chap. 6, where PH distributions are used to reduce the size of a stochastic model by modelling the behaviour of components based on their delay distributions.

Our goal was to provide our readers with the fundamentals to apply PH distributions in their own work. Of course, this means that we could only scratch the surface of the vast amount of work available on phase-type distributions. We would like to give a few pointers for further study to the reader interested in different aspects of the topic. For readers interested in the mathematical background, we recommend [683], where PH distributions were introduced, and the fundamental work in [251, 697, 698], which provides the basics for many of the closure properties and canonical forms used in the field. More information on PH fitting is available in the papers introducing the fitting tools we discussed here [446, 870, 880], and in [59], where a fitting tool for general PH distributions is described. [572] gives a survey of PH fitting tools that were available at the time and have not been considered here. An in-depth discussion of fitting heavy-tailed data is provided in [48]. Both [579, 683] provide good introductions to matrix-analytic methods. Lastly, with respect to random-variate generation we would like to also point out the approach in [157], where the authors present a method of generating random variates from Matrix-Exponential distributions, of which the PH distributions are but a subclass. The approach is particularly interesting because it differs from the methods discussed in this chapter in that it applies numerical inversion of the distribution function instead of playing the Markov chain.