Textbook for
TTM4110 Dependability and Performance with Discrete Event Simulation

Dependability and performance in information and communication systems

Fundamentals

Peder J. Emstad
Poul E. Heegaard
Bjarne E. Helvik
Laurent Paquereau

Department of Telematics, NTNU

Preface

This book provides the fundamentals for analysis, dimensioning, control and construction of ICT (Information and Communication Technology) systems. The topics covered include stochastic processes, basic traffic theory, dependability evaluation, and basic simulation techniques. The book covers the curriculum of the subject *TTM4110 Dependability and Performance with Discrete Event Simulation* given at Department of Telematics at Norwegian University of Science and Technology (NTNU).

The subject TTM4110 and a Norwegian edition of this book were created in 1999 as a joint work between Professor Peder J. Emstad, Associate Professor Poul E. Heegaard, and Professor Bjarne E. Helvik, all at the Department of Telematics, NTNU. The course was put together at a major revision of the study program and incorporated basic material up to then given in single courses covering statistics [1, 2], stochastic processes [3], teletraffic theory [4], simulation [5, 6, 7], and modeling and measurements [8]. Dependability issues were not covered at that time in basic courses and were added [9]. Efforts were made to present it all in a common framework and in a precise but engineering style. This English edition is a translation and revision of the Norwegian edition conducted mainly by PhD student Laurent Paquereau who has done an impressive work on translating, re-structuring, and proofreading. PhD student Marie Elisabeth Gaup Moe provided the first draft of Chapter 1. Poul E. Heegaard has maintained the Norwegian edition and acted as coordinator of this English edition. He has also translated and rewritten parts of the book in close cooperation with Laurent Paquereau.

This book is intended for undergraduate students and should provide the basic concepts and methods in preparation for specialization courses in teletraffic theory and dependable systems. Chapter 1 gives an overview of the subject and it is recommended to read it twice; first as an introduction but also as a summary after you have completed reading the book.

This edition is a draft. Comments and corrections are welcomed at pyse@item.ntnu.no tagged with [ttm4110][textbook] in the subject field. See the revision history for details.

Revision history

2016-07-01: Eleventh version.

- Minor rewriting of some of the definitions in Section 6.1.4.

2015-11-20: Tenth version.

- Correction of the solutions to the system of equations in Example 6.5 on page 176.

2015-06-26: Ninth version. Changes include:

- Corrections of typos reported by MSc student Einar Flobak.
- Various minor corrections and formatting changes.

2014-05-23: Eighth version.

- Corrections based on reports from MSc student Hans Henrik Grønsleth.

2013-06-21: Seventh version.

- Corrections of typos and minor formatting changes.

2012-06-03: Sixth version.

- Corrections of typos reported by PhD students Eirik Larsen Følstad, Maria Bartnes Line and Jonas Wäfler.

2011-06-08: Fifth version.

- Corrections of typos reported by PhD student Eirik Larsen Følstad.

2010-06-30: Fourth version.

- Corrections of typos based on reports from students following the course (including reports from PhD students Mark Stegelmann and Benedikt Westermann).

2009-06-22: Third version.

- Corrections of typos based on reports from students following the course.

2008-06-14: Second version. Changes include:

- Corrections (mostly of typos, including typos reported by PhD students Huaiyuan Ma and Anders Mykkeltveit and MSc student Thomas C. Vilarinho) and local rewriting and extensions by Laurent Paquereau, including Sections 2.2.4 and 6.5.
- Extended version of Chapter 9 written by Bjarne E. Helvik.

2007-05-01: First complete draft version available

Contents

1 Introduction

Technical systems are described using two types of characteristics: *functional* and *non-functional properties*, that is which functions are performed and how well they are performed, respectively. Let us illustrate this with an example outside the field of Information and Communication Systems (ICT-systems); a car. The primary function of a car is to transport people and goods from one place to another. Non-functional properties include the carrying capacity, the maximum speed, whether the car starts when needed, whether it transports people and goods to destination without breaking down or causing accidents. These are the *performance* (carrying capacity, speed) and *dependability* (the car starts and fulfills its function) properties of the system. Along with the price, they are rational arguments for comparing different designs. This is similar with ICT-systems. Let us take the example of a collaborative workspace. A collaborative software enables participants to work, communicate, share documents over local or remote networks. Performance and dependability characteristics for this service include:

- whether the service is up and running when needed, i.e. its *availability*; this includes the availability of all the components the system relies on.
- whether resources are sufficient in the network and at the end-user station, e.g. whether a request will be *lost* or not.
- how long it takes to establish a session, i.e. the *setup time.*
- whether the service is provided non-interrupted during the entire session, i.e. its *reliability*.
- whether the real-time requirements for user interactions (approx. 100 ms) are fulfilled, i.e. the end-to-end delay.
- whether the participants receive the data transmitted, i.e. the loss rate and the integrity of the received data.

When designing an ICT-system, the focus is mainly on the functions the system shall provide and how to implement them. However, in a real system, it is also crucial to consider non-functional properties such as dependability and performance. The non-functional requirements will have an impact on both the design and the cost of the system and determine its *usability.* The examples given above clearly illustrate this.

The purpose of this book is to give an introduction to the conceptual and theoretical fundamentals of dependability and performance of ICT-systems. Mathematical and software tools that can be used to analyze and dimension systems and network solutions are presented and basic issues are discussed.

This first chapter gives an overview of how dependability and performance connect with other features of a system and is organized as follows. Section 1.1 presents what determines the dependability and performance of a system. Section 1.2 introduces concepts and terminology related to dependability and performance, and Section 1.3 presents how dependability and performance modeling constitutes a part of the system development and dimensioning process.

1.1 What it is all about

Every dependability and performance evaluation of a system, either based on mathematical analysis, simulation or measurements, always relies on a *model* of the system. The model is an abstraction of the real or projected system. An evaluation cycle for a system is shown in Figure 1.1.

For regular dimensioning problems, extensively tested and recommended "standard models" exist so that the modeling phases of the cycle in Figure 1.1 are skipped. For instance, the Erlang's loss model described in Section 6.3.1 is a mathematical model to determine the congestion probability for a circuit-switched system. However, when using such a "standard system" one must be aware of the assumptions made which determine its applicability. Using a model when the assumptions do not hold may lead to very misleading results.

Some basic models can be applied to many different systems. For example, queueing models can be used to determine:

- the internal job flow and response times in a computer system with central processors, disks, caches, memories, input/output modules etc. ;
- the packet flow and end-to-end delay in a packet switched computer network;
- the probability that there is no repairman available when a module needs repair and the duration of the out-of-service period.

There is no standard recipe to elaborate a good model of a system and trade-offs must be made. On one hand a model must include sufficient details to represent the system, but on the other hand less important details must be left out to enable simulation in a reasonable time. Assumptions must be made so that the model can be expressed analytically, but at the same time one must ensure that the model still describes the real world. What all of this means will hopefully become clearer when studying this book. Note that one must always keep in mind that the results derived from a model should be valid for the real system, not just for the model!

In this section we look at the first phase of the modeling in Figure 1.1, that is the description of a system and its environments. The purpose is to identify the factors that must be included in the model. Some of them are illustrated in Figure 1.2. This phase is essential to construct a model or recognize a standard model. Assumptions, basic

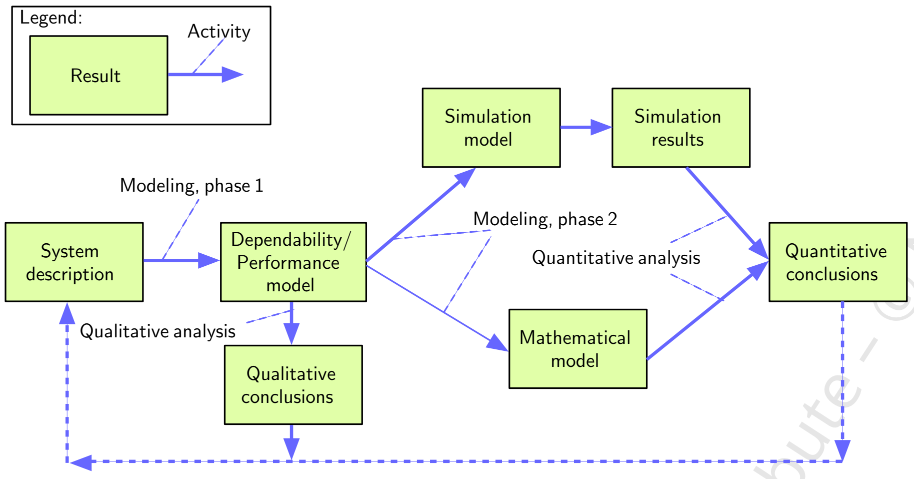


Figure 1.1: Modeling and analysis activities and results

models and examples, modeling techniques and quantitative analysis will be addressed in later chapters, after the mathematical and simulation fundamentals are established.

1.1.1 The system

A *system* can be defined as *a regularly interacting or interdependent group of items forming a unified whole* [Webster], where an item may be a system, a subsystem, or an atomic component. When performing a dependability and/or performance evaluation of a system, the aim is to identify the items (*system components*) that limit the dependability and/or the performance. The *structure* of the system reflects how these components interact. The interactions themselves are referred to as the *behavior* of the system. These three basic properties of a system will be detailed in the following sections.

A comprehensive description of all types of systems is of course not possible in this introduction. The objective is to give an idea of what needs to be taken into account in the modeling phase. In addition, remember that a system is a generic concept, and that a system can be studied at different levels (e.g. layers of the ISO OSI model) and at different levels of granularity (e.g. a router may be the system under evaluation, and in another evaluation the same router may be considered as a network component). Therefore, it is important to precisely define the system considered.

System components

An ICT-system is composed of *system components* of different types, for instance:

- processors with their processing capacity (measured in MIPS),

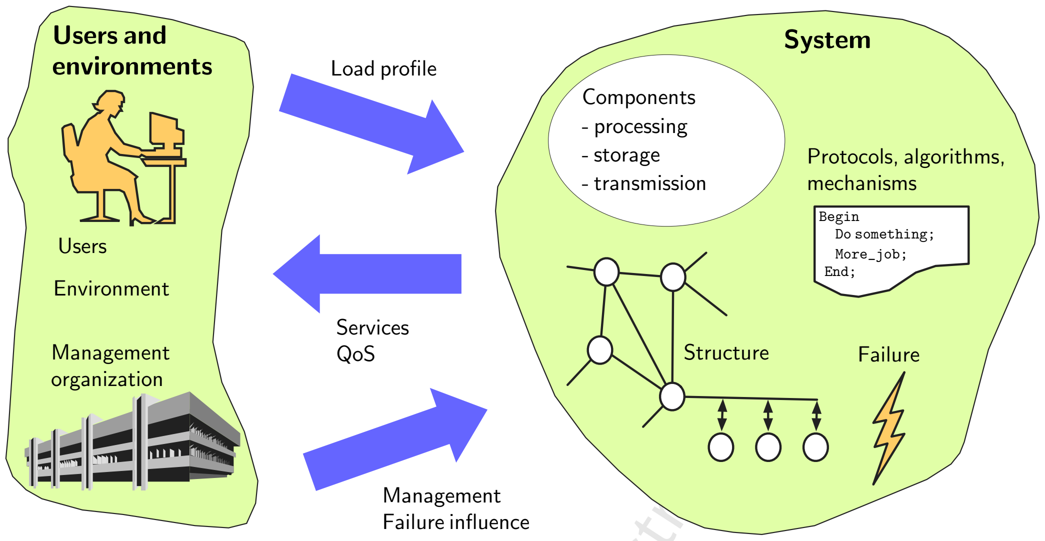


Figure 1.2: Illustration of the factors that must be considered in modeling, analysis and dimensioning of an ICT-system

- hard disks with a storage capacity (measured in Mbytes) and an access time (measured in s) and a transfer rate (measured in Mbytes/s), and
- transmission channels with a transmission capacity measured in kbits/s or Mbits/s. For instance:
  - buses inside a processor, or
  - communication links between nodes in a network.

System elements may also be aggregated, e.g. a router.

These system components and their capacities are the *resources* in an ICT-system. The amount of resources limits the system dependability and performance. The resources are what is utilized when the system is used.

For instance, consider a 2B+D ISDN access network. In this system there are three resources: the two B-channels of 64 kbit/s and the D-channel of 16 kbit/s.

Structure

The *structure* of a system indicates how the resources in the system must or should be utilized in order to deliver the service of which the dependability and/or performance properties are evaluated. Going back to the ISDN example above, if the performance of the 2B+D ISDN access system is evaluated with respect to establishing 64 kbit/s

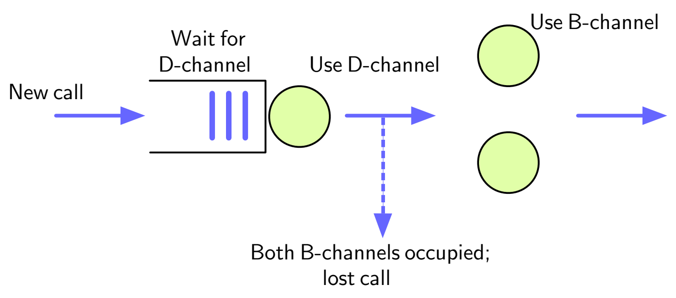


Figure 1.3: Simple model of a 2B+D ISDN access

circuit-switched connections, the D-channel is used for signalling first and then one of the B-channels is used for the communication. Assuming that there are no restrictions on the processing capacity, neither in the network nor at the end-user equipment, the simple model shown in Figure 1.3 can be constructed.

The structure extracted from the real system and included into dependability or traffic models usually mirrors the structure of the real system.

- In *dependability models*, the combinations of working system components that enable the system to deliver its services, determine the structure of the model. There is a class of dependability models that consider only the structure of the system, and not the dynamics of the system, see Section 7.3.
- In *performance models*, the combinations of system components that together deliver a service are also an element in the model. For example, in Figure 1.3, an arbitrary B-channel may deliver the service. It is also important to take into account how resources are used; sequentially or concurrently. In Figure 1.3 the sequential use of a D-channel and a B-channel is modeled. A circuit-switched connection throughout the whole network requires simultaneous reservations of a number of B-channels.

Note that the structure of a system may be:

- *physical*, e.g. how nodes in a network are interconnected.
- *logical*, e.g how units attached to a physical bus cooperate, as logical ring or hierarchically with a master and the others as slaves. As this example illustrates, the logical structure may differ from the underlying physical structure.
- derived from the physical and/or logical structures. In the above example, the bus and the units may have a serial structure with respect to dependability if all the units have to work for the bus-system to work.

Behavior

The third aspect of a system description which has to be considered when performing a dependability and performance modeling is the *behavior* of the system. Again it is necessary to simplify and make an abstraction of the behavior of the real system. Important aspects of the behavior that may be included in a model are listed below. Which to include depends on the system and on what is to be modeled.

1. *Queueing disciplines.* In most ICT systems, if all the resources are busy when one wants to use a system, one has to wait. This means that there is a logical or physical queue[1]. How this queue is organized may affect dramatically the performance of a system. Queueing disciplines may include:
   - queueing policy: FIFO (First In First Out), LIFO (Last In First Out), SJF (Shortest Job First), random, etc.,
   - priority: preemptive (with or without resume) or non-preemptive,
   - scheduling policy: round-robin, SRJF (Shortest Remaining Job First), etc.
2. *Protocols.* The main purpose of protocols is to provide rules for the different entities in a system/network to cooperate, e.g. :
   - to exchange information,
   - to control traffic flows,
   - to provide error control mechanism, and
   - to ensure consistency.

   Protocols may have mechanisms to control traffic flows failures, and hence are important aspects of the behavior of the system that must be taken into account when modeling the system.
3. Other *traffic mechanisms* apart from protocols, for instance:
   - routing algorithms; to optimize the use of network resources and cope with traffic variations and failures. Load sharing between processors and replication of functions are other mechanisms to ensure an efficient use of resources.
   - Call Admission Control (CAC) and User Parameter Control (UPC); to control the quality of the service delivered. CAC prevents that the system becomes overloaded so the service can be delivered as agreed on. UPC monitors and controls that users do not use the network more than they should.
4. *Fault-handling* mechanisms which encompass error detection, localization and isolation and various techniques to provide *fault tolerance* and automatic and manual *fault removal* (repair). Note that it is also important in a dependability model to include the possibility that the system does not behave as intended, e.g. that an error in the system is not detected.

[1]Queueing capacity may also be considered as a resource in some cases.

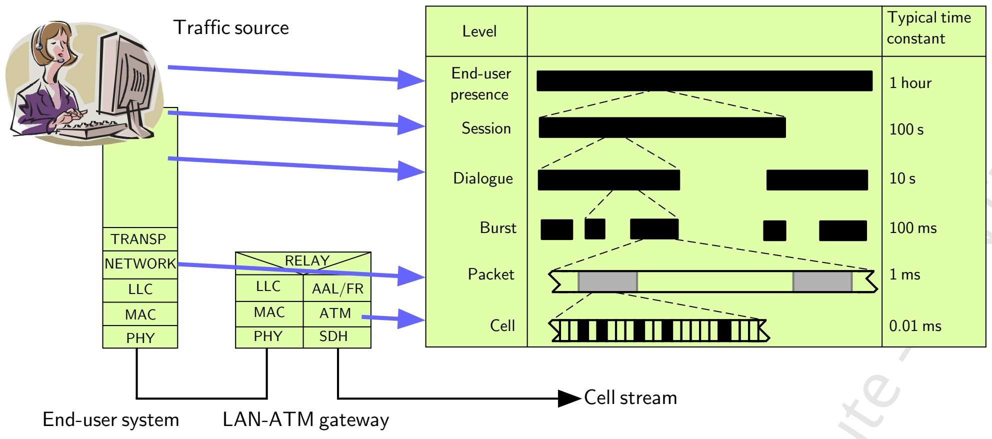


Figure 1.4: Illustration of end-user activities and equipment that determine the shape of the traffic at different time scales

1.1.2 The users and the environments

The description of the system itself and the identification of what to include in the model are essential, but not sufficient. How the environments influence the system and vice versa must also be considered.

Load profile

The load that is handled by an ICT-system is called the *traffic* and is determined by how often service requests are received and how long time it takes to serve the requests. A correct description of the traffic profile is critical for the validity of the evaluation of a system. If we go back to the example in Figure 1.3 and are interested in determining the probability of that a call is blocked, we see intuitively that the result depends on the number of users generating calls, how often they generate calls and how long the calls last. We can also easily imagine that users getting their calls blocked are likely to retry. In other words, there is a mutual influence between the state of the system and the traffic profile.

In traditional circuit-switched systems, the description of traffic profile is relatively simple because there is a single activity level; connected or not. In ATM and IP based transmission systems, the handling of the information at end-systems is far more complex, as illustrated in Figure 1.4. Various activities span over various time scales and various levels of abstraction. The traffic profile depends on:

- the end-user behavior at the highest levels. The end-user decides when to communicate, with whom and what media to use for the communication. The modeling

Table 1.1: A simple traffic matrix. The matrix gives the capacity needed in Mbit/s for a specific type of traffic at a specific time (the numbers are fictitious).

To/From	Oslo	Stockholm	Trondheim	Termination
Oslo	-	45	65	110
Stockholm	27	-	15	42
Trondheim	55	15	-	70
Origin	82	60	80	

of the user behavior (call attempts per day, mean holding time, etc.) is called the *traffic mix*.

- the content transmitted at the intermediate levels. For instance, speech encoders including silence suppression/voice activity detection produce a bursty traffic. Likewise, when transmitting an MPEG video sequence the amount of information transmitted at a particular instant depends on the level of details and motion, and on the type of frame, intra- or inter-coded.
- the hardware equipment, the physical media and the protocols at the lowest levels (creation, scheduling and transmission of packets). There are also interactions between the network and the end-user system, e.g. through sliding window and congestion control mechanisms, and retransmissions of lost packets.

As a result, the traffic, e.g. Internet traffic, may show almost fractal or self-similar properties.

The traffic observed in ICT-systems is in most cases an aggregation of a lot of individual flows coming from different users and depends on many factors. For instance:

- *the number of users*. Note that if there is a large number of users, it may be assumed that they are infinitely many. In this case and if it applies, *Palm-Khintchine's theorem* greatly simplifies the analysis, see Section 3.1.3.
- *the interests and the location* of the different users. See Table 1.1 for an illustration. Note that in this example the traffic is not symmetrical; the amount of traffic in one direction is different from the amount in the opposite direction. This is typically the case for web-surfing traffic.
- *the type of content*, e.g. speech, MPEG video, web content.
- *the variations in time*, not only at small time scales but also daily, weekly and yearly. An example of daily variations is given in Figure 1.5. In addition, it is interesting to try to give a prognosis of the long-term trend.

The prognosis of future traffic is important in a dimensioning process but is beyond the scope of this book. See [10] for an introduction. Traditionally (for telephony and

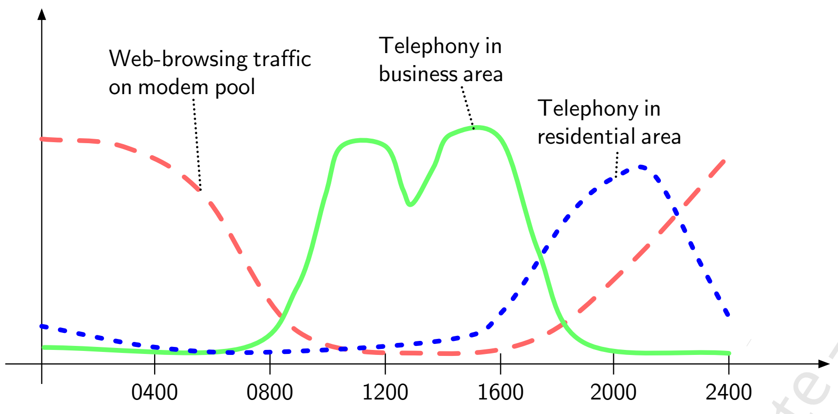


Figure 1.5: Examples of daily traffic variations

corporate data traffic) one could base the prognosis on demography, economy and industrial developments. Today, the development is characterized by disruptive events like the introduction of new capacity demanding services making good prognosis difficult to give. As an example, peer-to-peer services were not foreseen and have become a major volume generator of the Internet.

Operation and maintenance

The operation and maintenance support as well as logistics are important factors which must be considered in many dependability and performance problems and dimensioning studies.

In performance studies, the operation and maintenance must be taken into account only if they affect the traffic handling, which is the case in:

- *configuration control*, e.g. to dynamically change the available transmission capacity between nodes in the network depending on time of the day or on the offered traffic, and
- *traffic control*, e.g. to reject traffic at an early stage of an Intelligent Network (IN) service interaction when the network is already congested. Televoting where televiewers call to vote is a typical example. The vote counter may become overloaded and everyone attempting to reach it will use resources in the network, thus blocking also other users.

Actions (manual or automatic) taken depending on the load situation must also be considered. In some cases, the purpose of an evaluation is to identify or quantify the impact of different operation and maintenance strategies and traffic load control mechanisms, e.g traffic shaping or load sharing.

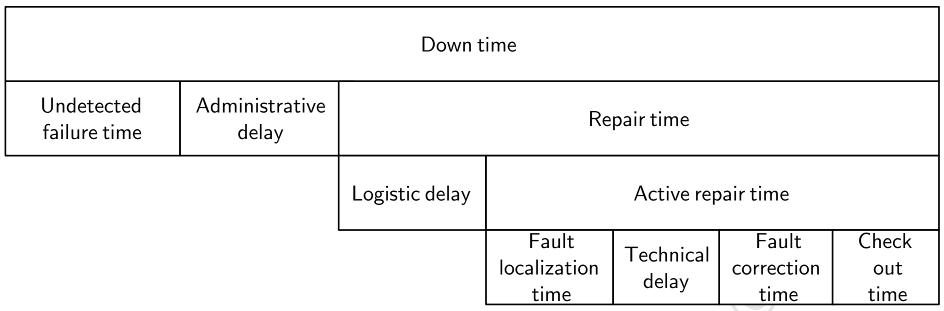


Figure 1.6: Partition of the system down times which require a manual repair according to ITU-T E.800 [11]

Dependability studies must always take into account operation and maintenance. Failures can have their cause in the system itself or in the environments, see Section 1.2.2. The traffic load and the maintenance activities influence the failure rate.

Operation and maintenance functionality includes:

- *configuration control*, e.g. to ensure that there is enough spare transmission capacity between network nodes so that traffic can be re-routed in case of a link failure without affecting the performance/carried traffic, and
- *fault-handling*, e.g. to detect when a router fails and localize the part of the router that has failed, re-route traffic, repair (*fault removal*) the faulty part, restart and test the repaired part, and finally route traffic again through the router.

When dealing with physical failures, the maintenance strategy determines the consequences these failures have on the dependability of the system and the services it delivers. Several factors come into play, see Figure 1.6. In addition to the time it takes to repair the failure (active repair time), the time it takes for the repairman, the new/spare parts and other required equipment to come (logistic delay), and the time it takes to initiate the repair (administrative delay) must be accounted for. Maintenance is costly and therefore strategies are elaborated to minimize this cost. However, to some extent, a reduction of the maintenance cost is at the expense of the dependability. For instance:

- minimizing the spare parts in stock,
- delaying maintenance until the next failure or until a more convenient time, e.g. the next working day (this may be possible if the system is made fault tolerant and can continue delivering its services even though some elements have failed).

A common mistake when performing dependability modeling and dimensioning is to rely on a too simple description of the operation and maintenance aspects.

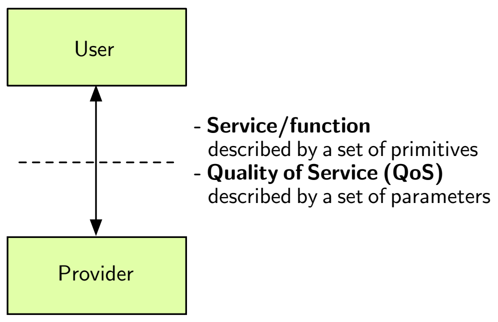

User
- Service/function
described by a set of primitives
- Quality of Service (QoS)
described by a set of parameters
Provider

Figure 1.7: Generic user/provider service model

1.2 Concepts and terminology

This section introduces concepts and terminology related to dependability and performance. Some of the concepts will be further discussed throughout this book. There is a large body of standards and definitions of concepts and terms within the area. They are not necessarily consistent with each other, and often intended for specialized use with regard to certain topics, or certain types of organizations. One example is the ITU-T recommendation E.800 "Terms and Definitions Related to Quality of Service and Network Performance including Dependability" [11], that focuses on connection oriented services in public telecommunication networks from the point of view of network operators[2]. Other important concepts and definitions of terminology can be found in [12] for layered systems, in [13] for establishing QoS requirements toward the end-user and in [14] for the dependability and security (trustworthiness) area. The following presentation is not based on one specific reference, but aims at giving a broad and general introduction.

1.2.1 Quality of service

Quality of service (QoS) is a notion that has received an increased attention with the introduction of new technologies in networks. Since quality is a general concept, the interpretations of QoS are many. This is also the case within the ICT-community, leading to confusion and misunderstandings. Three different interpretations are listed below.

1. *The delivery of a service in accordance with its specification.* This definition assumes that each service has its own set of QoS parameters with corresponding values, for instance, with respect to the example given in the beginning of this

[2]The general parts of E.800 have been adopted by IEC (International Electrotechnical Commission) as the terminology standard IEV 191.

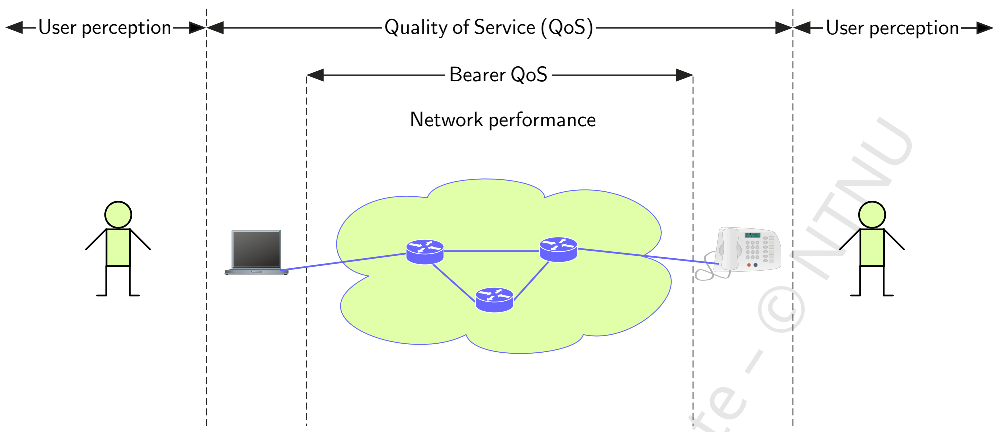


Figure 1.8: Distinction between network performance and QoS in ITU-T E.800 [11]

chapter, an availability larger than 99.9%, a blocking probability less than 1% and a setup time less than 200 ms etc. The quality depends on to what extent these requirements are met. This is an operational definition that enables dimensioning a system in accordance with the specified requirements. Additional specifications, regarding how to measure these parameters, enable a quantitative evaluation of the service by both the user and the provider.

This definition relates to a generic user/provider service model as shown in Figure 1.7. The functional requirements are described in terms of a set of *service primitives*. The quality of the service delivered is described by a set of *quality of service parameters*. The most relevant quality of service parameters for dependability and performance are detailed in sections 1.2.2 and 1.2.3, respectively.

2. *The end-user satisfaction with a service.* This definition is used in [11]. If it is to be used in an engineering context it has to be detailed and concretized and will thus resemble the previous definition. Note that this definition relates the QoS only to what the user experience. In E.800 [11], *network performance* is used to denote the properties of a network, a sub-network or a single network element. See Figure 1.8. We do not make this distinction in this book.
3. *The existence of mechanisms (in the network) for controlling the use of the different resources.* This definition has its origin in the efforts of using the Internet, i.e. the IP protocol suite, to handle service integrated traffic. Traditionally all the resources in the Internet have been equally shared between the users. Today the Internet is used to provide all kinds of services, including services with real-time constraints like speech. Such services make it necessary to control the use of the transmission capacity so that they get sufficient capacity and small enough delay to be usable. This can be achieved by using different priorities (e.g. DiffServ) and reservation mechanisms (e.g. RSVP and IntServ). That is, the network provides QoS by means of service differentiation.

> Such a definition of the QoS should be avoided as it mixes up the functional properties (service primitives) and the non-functional properties.

In the following we stick to the first type of definition.

Definition

Quality of Service (QoS)

Degree of compliance of a service to the agreement that exists between the user and the provider of this service.

In this context, a *user* is an entity that uses a service provided by another entity as illustrated in Figure 1.7, but is not necessarily an end-user of the service. For example, a transport protocol is the user of the services provided by a network protocol, and an application process in a node is a user of the services provided by the middleware of this node.

A *service* is a set of functions (*service primitives*) that are offered on an interface between the user and the provider. Note that this is not necessarily a physical interface. A *QoS parameter* is a (random) variable that characterizes the service.

Compound systems

With the definition given above it is possible to decompose an ICT-system into a number of users and providers according to the layering and the segmentation of the network. An example of layering is the different layers in the ISO OSI model. In this model a specific layer is a provider of services for the above layers and a user of services provided by the underlying layers. The description of the activities in Figure 1.4 illustrates this. By segmentation it is referred to physically separated parts of the system (horizontal partitioning) like the end-user system, the access network, the transport network belonging to a network operator A, the transport network belonging to a network operator B, the server system belonging to a service provider C, etc. The simple model in Figure 1.7 can be extended to include several horizontal layers and vertical segments as illustrated in Figure 1.9. A user-provider graph, like the one in Figure 1.9, will not always have a tree- structure and the interfaces in such a model will not always be physical.

If the dependability and the traffic mechanisms and properties of all the provider/user associations in are fully known, it should be possible, in principle, to derive the quality of service experienced by the end-user. For example, it should be possible to study the effect of the bit error rate on a transmission link (leaf node provider at the physical layer) on the quality of the speech[3] perceived by a human end-user (the user of services

[3]For instance, the mean opinion score (MOS) provides a numerical measure of the quality of speech.

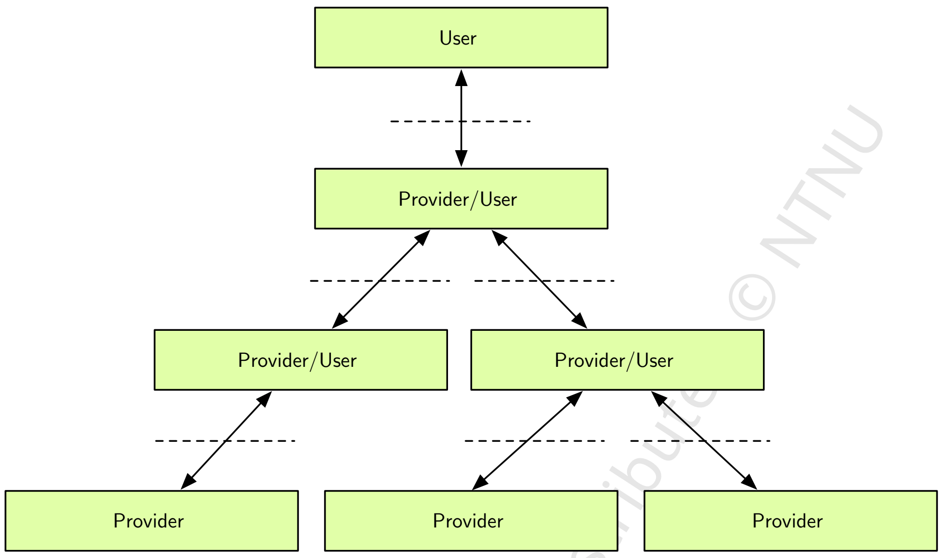


Figure 1.9: User/provider graph

at the application layer). This depends on how the transmitted information is handled by the provider/user elements in the network, e.g. :

- Forward Error Correction (FEC)[4],
- detection and retransmission of erroneous/lost packets. Note that this introduces a delay that may have a significant impact on the quality of speech.

However, it is in general a non-trivial task to derive high level QoS measurements based on measurements done at lower layers.

Commercial application

In the above paragraphs, we have only considered QoS from a technical point of view, i.e. as a characteristic of a service delivered from one technical entity to another. (The human end-user is an exception here.) QoS is also very important in a commercial context. The user and provider in Figure 1.7 may be two commercial actors, where the user buys a service (or a set of services) from the provider. The user may be for example:

- a private end-user,
- a company (e.g. a bank),

[4]FEC (Forward Error Correction) means that the information bits are encoded using an error correcting code, e.g. a Hamming code, that enables to correct up to a certain number of bit errors in a block of information.

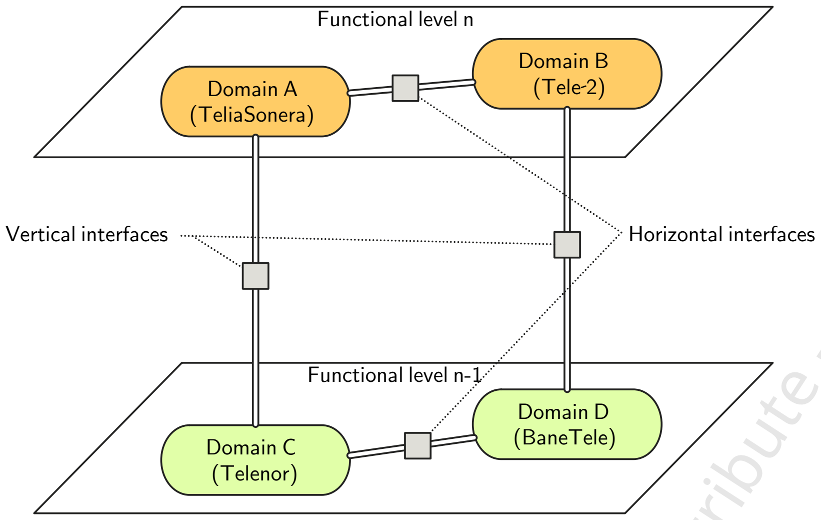


Figure 1.10: Interfaces between different market players (trademarks are only given for illustration purposes)

- a service provider that buys transport services from a network operator,
- a network operator that has an interconnection agreement with another network operator.

Requirements on the QoS are given in terms of tolerance levels on the QoS parameters. These levels are specified in a *Service Level Agreement (SLA)*. The SLA may include:

- the service or set of services (with it/their service primitives) that is/are to be delivered,
- the QoS parameters, see Sections 1.2.2 and 1.2.3, how they are measured and the tolerance levels they are to be kept within,
- the amount of traffic the user can transmit,
- the measures taken if the QoS parameters exceed the tolerance levels (e.g. discount if delays or error rates exceed the specified values).

A service level agreement is associated with an interface in the network. We distinguish between:

- horizontal interfaces, between network segments at the same level but belonging to different actors, and
- vertical interfaces, between actors that deliver services at different functional levels.

For example, domains C and D in Figure 1.10 belong to two different network operators, while domain A and B are two providers of end-user services.

1.2.2 Dependability

Definition

Reliability is a part of our everyday language. A reliable person is a person that can be relied on to do what is expected from him/her, and if he/she does anything more than what is agreed upon this does not have any unwanted side effects. *Dependability* has a corresponding meaning in a technical context. The term *dependability* has been chosen because the term *reliability* was given earlier a very specific technical meaning. The definitions given in this section are all taken from [14].

Dependability

Trustworthiness of a system such that reliance can justifiably be placed on the service it delivers.

Dependability is a high-level concept. In addition to the *dependability attributes* which will be discussed later, dependability also encompasses the *impairments* that could affect the trustworthiness of a system, and the *means* to attain dependability. This is illustrated in Figure 1.11.

Threats to dependability

Failure

Deviation of the delivered service from the compliance with the specification. Transition from correct service to incorrect service (e.g. the service becomes unavailable).

Error

Part of the system state which is liable to lead to a failure.

Fault

Adjudged or hypothesized cause of an error.

The relation between these concepts is illustrated in Figure 1.12. An electromagnetic pulse (fault) results in flipping of a bit in a data register (error). When this register is accessed, a wrong result is returned to the user (failure). Another example is the software engineer who writes an incorrect code and thereby introduces a logical fault into a software module. This incorrect code is a (dormant) fault embedded in the system. Certain input values will activate the fault and there will be an error in the

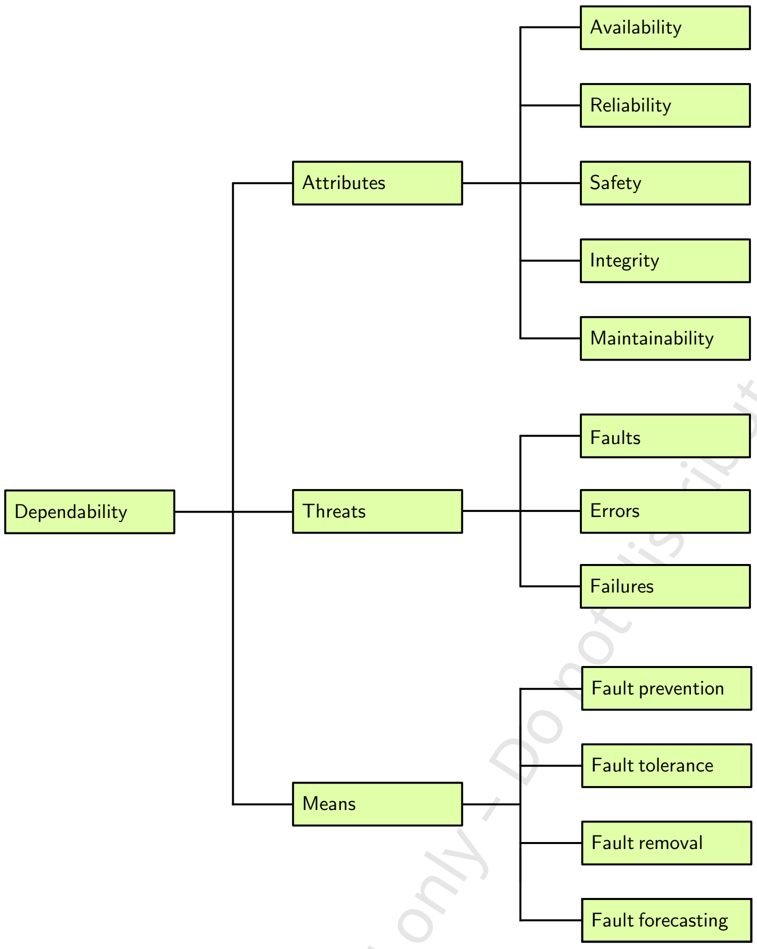


Figure 1.11: The dependability tree

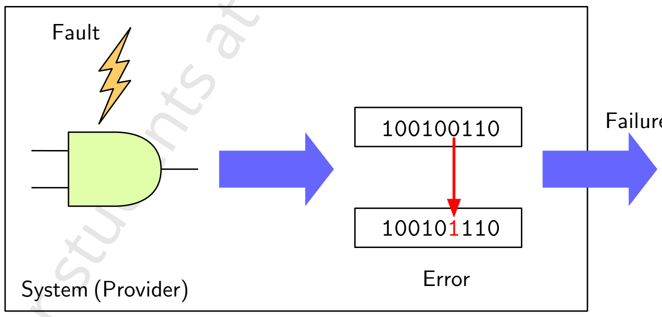


Figure 1.12: Relation between fault, error and failure

system, e.g. a broken link in a linked list. Services delivered by the software module that use this list may later crash or produce incorrect output (failure).

The conceptual distinction between fault, error and failure is very important. For example, the two basic approaches to achieve a dependable system are:

- *fault prevention*, i.e. to prevent the occurrence or introduction of faults, and
- *fault tolerance*, i.e. to prevent that errors cause failures or, in other words, to deliver a correct service despite the presence of faults.

See Section 7.1.2 for more details.

Types of faults

This book focuses on modeling and analysis of the dependability and performance of ICT-systems, and not on the system development process. Correspondingly, the focus is on failures and service outages (periods of delivery of incorrect service) rather than on the primary causes of the lack of dependability (faults and errors). However, it is important to be aware that faults vary in their causes and persistence. This section gives a brief overview of the different types of faults and how they can be classified, see Section 7.1.1 or [14, 9] for details.

- *Physical faults.* These are the "classical" faults, e.g. physical wear out of components.
- *Transient faults.* These faults are present only for a short period of time and no physical change occurs in the system, e.g. external disturbances like electromagnetic interference and radiation.
- *Intermittent (sporadic) faults.* These faults come and go, for instance due to a hardware components operating close to its tolerance limit, e.g. a gate that receives only sometimes (and not always) a satisfactory signal level.
- *Design (logical) faults.* These are human made faults during specification, design and implementation of a system. Design faults may be found both in the hardware and software, as well as in interactions between hardware and software.
- *Interaction or operational faults.* These are accidental faults made by humans operating or maintaining a system, e.g. misconfiguration of the system, typos in configuration files.
- *Faults caused by the environment.* These faults originate outside the system boundary, e.g. power outages or fire.

Both the relative and absolute frequencies of failures, caused by the faults described above, might vary from system to system. Note, however, that physical faults rarely occur, while logical and system interaction faults are relatively frequent. As an illustration, consider the number of failures of a personal computer that are due to:

- physical faults (a hardware component needs to be changed),
- the operative system or an application software,
- tampering with the system, new installations etc.

Details on how to tolerate and repair different kinds of failure are outside the scope of this book. For a comprehensive introduction, see for example [14] or [9]. However, note that different types of faults give very different failure times. The repair of a physical failure may take weeks, including the time spent obtaining the new parts, while a restart (due to a transient or logical fault) may only take tens of milliseconds for a small sub-system executed from a code in a ROM. It is important to keep this in mind when building dependability models and analyzing system properties based on these models.

System times

The behavior of a system in operation is illustrated in Figure 1.13.

The system is either:

- working (up), i.e. it delivers its services according to the specification, or
- failed (down), i.e. the services are not delivered or delivered with content or timing failures.

$I(t)$ is a function of time that describes the behavior of the system.

$$I(t) = \begin{cases} 1 \text{ if the system is working at time } t \\ 0 \text{ otherwise} \end{cases} \tag{1.1}$$

The system times illustrated in Figure 1.13 are listed below. At $t = 0$ the system is new or can be considered as new (no degradation or faulty components).

T_{FF} *Time to First Failure.*

T_{CF} *Time to first Catastrophic Failure.*

T_{BF} *Time Between Failures.* In general, it can *not* be assumed that times between failures are identically distributed or independent during the lifetime of a system. This applies also to T_U and T_D.

T_U *Up Time*: time from the service of a system is restored after a failure and until the next failure, i.e. duration of the working period.

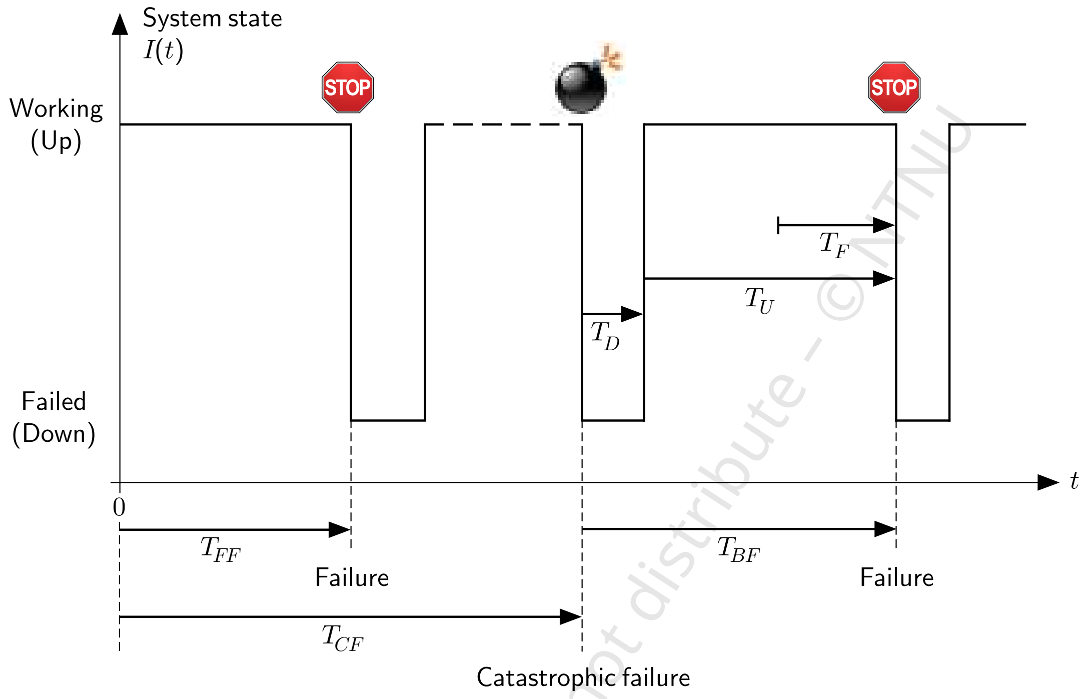


Figure 1.13: Illustration of system times

T_D *Down Time*: time from the system ceases to deliver its service until the service is restored, i.e. the system outage[5].

T_F *Time to Failure*: time from a random instant when the system is working until it fails.

System times are random variables, see Section 2.2. Given that they admit an expected value, see Section 2.3.2, the following symbols are used:

$MTFF = \mathrm{E}(T_{FF})$ *Mean Time to First Failure.*

$MTCF = \mathrm{E}(T_{CF})$ *Mean Time to Catastrophic Failure.*

$MTBF = \mathrm{E}(T_{BF})$ *Mean Time Between Failures*[6].

$MUT = \mathrm{E}(T_U)$ *Mean Up Time.*

$MDT = \mathrm{E}(T_D)$ *Mean Down Time.*

$MTTF = \mathrm{E}(T_F)$ *Mean Time To Failure.*

[5] T_D strongly depends on the fault causing the failure.
[6] $MTBF$ assumes that the system can be repaired. It is sometimes incorrectly used for $MTFF$.

Availability

Availability

Ability of a system to provide a set of services at a given instant of time or at any instant within a given time interval.

The *asymptotic availability* is the most common availability measure[7] and is denoted A. It is assumed that the system has reached its steady state and A is the probability of finding the system in a working state at a randomly chosen time in the future. Formally:

$$A = \lim_{t \to \infty} P(I(t) = 1) \tag{1.2}$$

From the definition of the availability we have:

$$A = \frac{MUT}{MDT + MUT} = \frac{MUT}{MTBF}. \tag{1.3}$$

The asymptotic availability of a system has typically a value between 0.9 (home computer without any support) and 0.99999 (network component with high availability, e.g. a telephone exchange).

Since the (asymptotic) availability is close to one, the *unavailability* U, i.e. the probability of being unavailable, is often used instead.

$$U = 1 - A = \frac{MDT}{MTBF} \tag{1.4}$$

The unavailability of a system therefore typically lies between 10^{-1} and 10^{-5}.

Other availability measures includes the *instantaneous availability*, or *pointwise availability*, which is the probability that the system is working at a given instant t

$$A(t) = P(I(t) = 1) = \mathrm{E}(I(t)) \tag{1.5}$$

and the *interval availability*, or *mean availability*, which is the average availability over a specified period.

$$\begin{aligned} A(t_1, t_2) &= \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} A(t)\,\mathrm{d}t \\ A(t_1, t_2) &= E\left(\frac{1}{t_2 - t_1} \int_{t_1}^{t_2} I(t)\,\mathrm{d}t\right) \end{aligned} \tag{1.6}$$

These measures converge to the asymptotic availability, given that it exists.

$$A = \lim_{t \to \infty} A(t) = \lim_{t \to \infty} A(t, t + \tau) \tag{1.7}$$

[7] we distinguish between a property, e.g. the availability, and the different measures of this property.

The corresponding unavailability measures are defined as

$$U(\dots) = 1 - A(\dots) \tag{1.8}$$

Remember that the availability measures are probabilities and expected values, not what is observed at an instant or during a period of time, $\tau^{-1} \int_{t_1}^{t_1+\tau} I(t)\,\mathrm{d}t$, which are random variables.

The interval (un)availability is often given as an accumulated down time per time unit, e.g. 3 minutes of down time per year or two hours accumulated down time per 40 years. (These values were the design objectives for computer controlled telephone exchanges (SPC systems) that were developed at the end of the 1960's. In 1985 most of the major manufacturers reported that the operational experience of their systems showed that these objectives had been reached).

Reliability

Reliability

Ability of a system to provide uninterrupted service.

New system If the service is received from a system which is new, or considered as new, when its usage starts, e.g. a satellite, the reliability is determined by the time to first failure T_{FF}.

The *reliability function* is defined as

$$R(t) = P(T_{FF} > t) \tag{1.9}$$

That is, the reliability function is the probability that the system provides uninterrupted service during $[0, t]$.

Another measure of the reliability of the system, which contains less information, is the mean time to first failure, $MTFF$. If $R(t)$ decays to zero sufficiently fast, i.e. if $\lim_{t\to\infty} t\,R(t) = 0$,

$$MTFF = \int_0^\infty R(t)\,\mathrm{d}t \tag{1.10}$$

System in steady state If the system is not new but in steady state with respect to failures and restorations of service, the reliability is given by the time to failure T_F. For instance, when a telephone user makes a call, the network is in steady state, and the reliability is given by the probability that the call terminates before a failure occurs.

The reliability function is then defined as

$$\tilde{R}(t) = P(T_F > t) \tag{1.11}$$

and if $\lim_{t\to\infty} t\,\tilde{R}(t) = 0$, the mean time to failure, $MTTF$, is

$$MTTF = \int_0^\infty \tilde{R}(t)\,\mathrm{d}t \tag{1.12}$$

Safety

The impact a failure has on the system environments is called *failure effect*. The failures of a system are divided into two classes with respect to the failure effect:

- *benign failures* , and
- *catastrophic failures* .

Whether a failure is benign or catastrophic depends on the given application and on the consequences it has. The same failure may lead to very different consequences. For instance, if a signal and control system for railways fails and causes a derailment or a collision, the failure effect is worse (catastrophic failure, accident) than if it only leads to some trains stopping (fail-safe, standstill). Another example can be fetched from ICT-systems; if a communication system breaks down (e.g. a telephone exchange), the consequences may be catastrophic if it was used for controlling the air traffic or for an emergency call center, or benign if it was only used for ordinary phone calls.

Safety

Ability of a system to provide service without the occurrence of catastrophic failures.

With respect to safety, the benign failures “does not count”. The safety is measured by the safety function, $S(t)$. $S(t)$ is the probability that no catastrophic failure has occurred until and including time t. Thus, safety may be regarded as reliability with respect to catastrophic failures.

$$S(t) = P(T_{CF} > t) \tag{1.13}$$

The mean time to catastrophic failure exists if $S(t)$ decays to zero sufficiently fast, i.e. if $\lim_{t\to\infty} t\,S(t) = 0$, and is given by

$$MTCF = \int_0^\infty S(t)\,\mathrm{d}t \tag{1.14}$$

1.2.3 Performance

Similarly to reliability, performance is also part of our everyday language. In the context of ICT-systems, it is defined as follows:

Performance

Ability of a system to provide the resources needed to deliver its services.

In other words, the performance depends on the amount of resources in the system and their utililization. Going back to the car example given in the beginning of this chapter, the performance is associated to the power of the car (resource), its weight, the power transmission, the roadability, etc.

Capacity

Capacity

Maximum load a system can handle per time unit.

The capacity is a property of a system, and not of the service it delivers. The capacity available to a user, or a group of user, for its services is called the *throughput* and will be discussed later in this section.

The capacity is denoted C and its unit varies according to the type of system considered, and how the load is defined for this system. For instance:

- the capacity of a transmission channel (service provided at the physical level) is the number of bits transferred per second (datarate).
- the capacity of a routing function in an IP-router is a number of packets per second.
- the capacity of a processor is often given in
  - MIPS (Million Instructions Per Second) for a control system, and
  - MFLOPS (Million FLoating point Operations Per Second) for a calculation system.
- The capacity of a database is a number of transactions per second.

Note that some systems can *never* handle a load close to their capacity because of the way the resources of the system are managed. An example of such system is an Ethernet where the medium (resource) is shared by several users which contend for access using CSMA/CD[8].

[8]Carrier Sense Multiple Access / Collision Detection

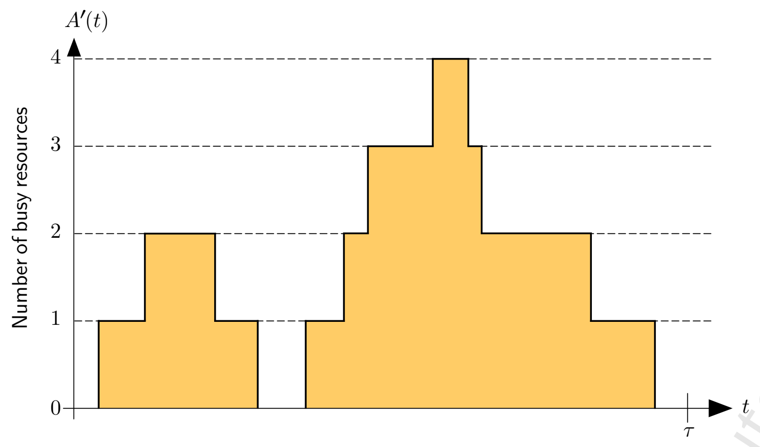


Figure 1.14: Example of utilization of resources. The carried traffic during the time interval $[0, \tau]$ is the average of $A'(t)$, $A' \approx 1{,}9$ Erlang.

Traffic and utilization

Instantaneous carried traffic $A'(t)$

Number of busy resources in a pool of resources at a given instant of time t.

The pool of resources may be for example a number of channels in a circuit-switched system, or a number of printers in a printer pool.

Carried traffic A'

Average number of busy resources in the time interval $[0, \tau]$.

$$A' = \frac{1}{\tau} \int_0^\tau A'(t)\,\mathrm{d}t. \tag{1.15}$$

The unit for the traffic is the *Erlang* in honor of the Danish mathematician A. K. Erlang, who was the founder of traffic theory. This unit is dimensionless. The total traffic carried in a time period of length τ is a *traffic volume*, and is measured in *Erlang-hours*. See Figure 1.14 for an illustration.

If the n resources of a system are equally loaded, the traffic per resource, or *utilization* of a resource, is the portion of time the resource is occupied and is given by $\rho = A'/n$. A resource can at most carry one Erlang.

Offered traffic A

Traffic that would be carried if the number of resources were infinite.

The offered traffic is greater or equal to the carried traffic. The difference between the offered traffic and the carried traffic is the lost traffic.

Lost traffic A''

Difference between the offered traffic and the carried traffic.

Loss occurs when a system does not have sufficient resources to serve the incoming service requests. It is often quantified by the loss, or blocking, probability.

In circuit-switched communication systems, the blocking probability is called *congestion*. We distinguish between:

- *Call congestion B:* fraction of all call attempts which are lost because they see all the resources busy (user-perceived quality of service)
- *Time congestion E:* fraction of time all the resources are busy.

Call congestion and time congestion may differ depending on the arrival process. See Section 6.3.

Loss may also occur in connectionless systems, e.g. *packet loss* in an IP network, and in the connection phase for asynchronous virtual connections, e.g. *cell loss* in ATM networks.

The dimensioning of an ICT-system is based on the offered traffic during the busy hour. The *busy hour* is defined as the 60 minute period of the day (defined with a 15 minute granularity) with the highest carried traffic on average over a long period of observation.

Inside a complex system, we distinguish also between:

- the *primary traffic*, which is generated by the users, and
- the *secondary traffic*, or signalling and control traffic, e.g. utilization of transmission channels for signalling between network nodes, transmission of traffic measurements internally in the network, updating of routing information, etc.

All these concepts are illustrated in Figure 1.15 and a more comprehensive presentation is given in Chapter 6.

Throughput

Throughput

Portion of the system capacity that is utilized by the users.

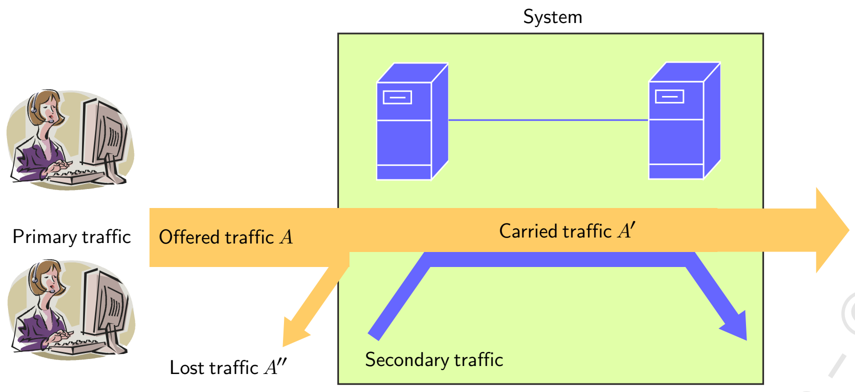


Figure 1.15: Relation between the different traffic concepts

The throughput for all users of the system is denoted ζ and is given by the product of the capacity and the utilization.

$$\zeta = \rho \cdot C \tag{1.16}$$

The throughput for a single user is the portion of system capacity utilized by this user. This is an important measure of quality of service for connectionless communications, where the users can not reserve resources and have no guarantee on the number of resources available in the network.

Delay

Delay

Time it takes to complete a service.

Several contributions to the delay can be identified. For instance, in a store-and-forward network, the end-to-end delay for packets includes the delay between nodes in the network (propagation delay) and at nodes (waiting time and service time) as illustrated in Figure 1.16.

System times

Waiting time W

Accumulated time a service request is pending for service in a system.

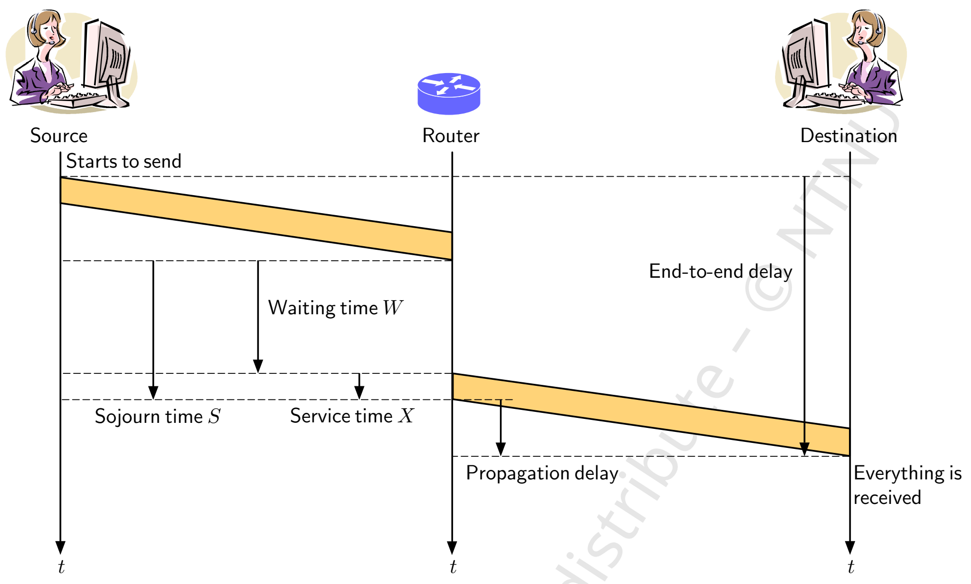


Figure 1.16: Contributions to the delay in a store and forward system

In the example in Figure 1.16, the waiting time is the time from when the packet (service request) is completely received by the router until the router starts serving it. In this example, the packet is served continuously. However, some queueing disciplines may result in discontinued service, e.g. preemptive priority. The waiting time is then the accumulated time the packet is not processed while being in the system.

In this example, time elapses from the reception of the beginning of the packet until it is completely received. In other types of networks, where forwarding of a packet can start as soon as the header is received (cut-through switching), e.g. in some ATM switches, the waiting-time starts when the header is fully received. In many cases the time it takes to receive a service request is disregarded.

Common measures of the waiting time are the *cumulative waiting-time distribution*:

$$F_W(t) = P(W \leqslant t) \tag{1.17}$$

and the *expected waiting-time* $\mathrm{E}(W)$.

Service time X

Accumulated time a service request is served by a system.

A service request is served when it utilizes resources in the system. The service time depends on the service request to serve, for example:

- the duration of a circuit-switched connection (call duration in telephony),

- the volume of information to be transferred on a packet switched communication channel,
- the number of pages that is to be printed on a printer,
- the number of instructions needed to process a call,
- the amount of data to be read from a database.

Note that in all these examples, except the first one, the service time also depends on the capacity C of the system.

Common measures of the service time of a system are the *cumulative service time distribution*

$$F_X(t) = P(X \leqslant t) \tag{1.18}$$

and the *expected service time* $\mathrm{E}(X)$.

Sojourn time S

Total time a service request is in a system.

Thus, the sojourn time is the sum of the waiting time and service time

$$S = W + X \tag{1.19}$$

and the expected sojourn time is, see Section 2.3.7,

$$\mathrm{E}(S) = \mathrm{E}(W + X) = \mathrm{E}(W) + \mathrm{E}(X) \tag{1.20}$$

End-to-end performance

Until now we have only considered the delay introduced by a single subsystem, e.g. the router in Figure 1.16. However, one is also often interested in the total delay from sender to receiver (end-to-end delay), for instance for real-time services. The end-to-end delay includes the system times in all the subsystems (all the nodes in the network) that take part, as well as the delay introduced by the physical transmission (propagation delay). The propagation delay depends on the physical distance and on the propagation speed (the speed of light in air/vacuum and approximately 2/3 of the speed of light in optical fibers). For connections traversing geostationary satellite links, the one-way propagation delay is typically on the order of 300 ms. This yields to a round-trip time on the order of 600 ms which may be prohibitive for some real-time services.

The end-to-end delay for asynchronous transmissions is the time from the first bit in a PDU[9] leaves the sender until the last bit in the PDU is received by the receiver. For synchronous transmissions, it is the time from when one bit is sent until it is received.

[9]Protocol Data Unit

For real-time services using asynchronous transmissions, the variation of delay, or *jitter*[10] is also of importance. The jitter determines, among other things, the size of the jitter buffer needed at the receiver side to get a synchronous playback of sound and image. There is no precise definition of the jitter, but it could be characterized by the standard deviation, see Section 2.3.2, of the end-to-end delay. Another important measure related to the end-to-end delay is the *bandwidth-delay product*. It is the product of the transfer rate and the minimum round-trip delay. It gives the maximum amount of unacknowledged data transmitted by the sender. This is relevant for protocols which need to keep the transmitted data in a buffer until an acknowledgment is received; for instance TCP (Transmission Control Protocol).

In addition to the delay, there are many other end-to-end QoS measures of interest. A summary of the most important ones for different types of bearer services and phases in connection oriented communication is given in Figure 1.17. Some dependability measures are included as well.

1.2.4 Relation between dependability and performance

In Sections 1.2.2 and 1.2.3 we have treated dependability and performance properties separately. However, these are not independent of each other.

A system in which failures never occur but without sufficient resources to meet the performance requirements is not considered to be in a working state. Whether the performance requirements of a system are met or not determines whether the system is in a working state or not. For example, a best-effort transport service, as in the traditional Internet, is up as long as the throughput is greater than zero. This can be formalized by setting reference performance values and using them for deciding whether the service is up or not, as illustrated in Figure 1.18.

Relations between some properties of a system are shown in Figure 1.19. In a large and complex system, e.g. a telecommunication network, failures and repairs of parts of the system are common events. These events constitute the behavior of the system with respect to dependability and determine the resources available for traffic handling and the QoS experienced by the users of the system. For example, if a transmission link between two routers in the Internet goes down, the routing protocol tries to find new routes for the traffic (if such routes exist). The re-routing is likely to result in a decrease of the performance experienced by the users (smaller throughput and/or a larger delay). Whether this degradation is acceptable or not determines whether the transport service is considered to be up or down. In the traditional Internet, the service is considered to be up as long as there is connectivity and some capacity available to the users while, in the traditional (virtual) circuit switched telecommunication network, almost no capacity reduction is accepted before the service is considered as down.

[10]This jitter must not be confused with the bit transmission jitter at the physical layer.

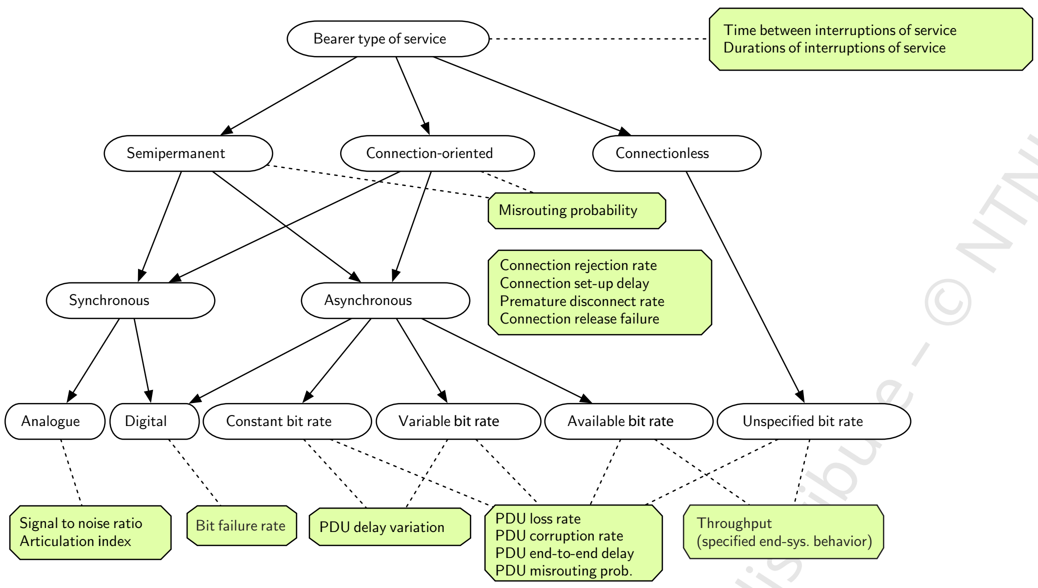


Figure 1.17: End-to-end QoS measures for bearer services in a communication network

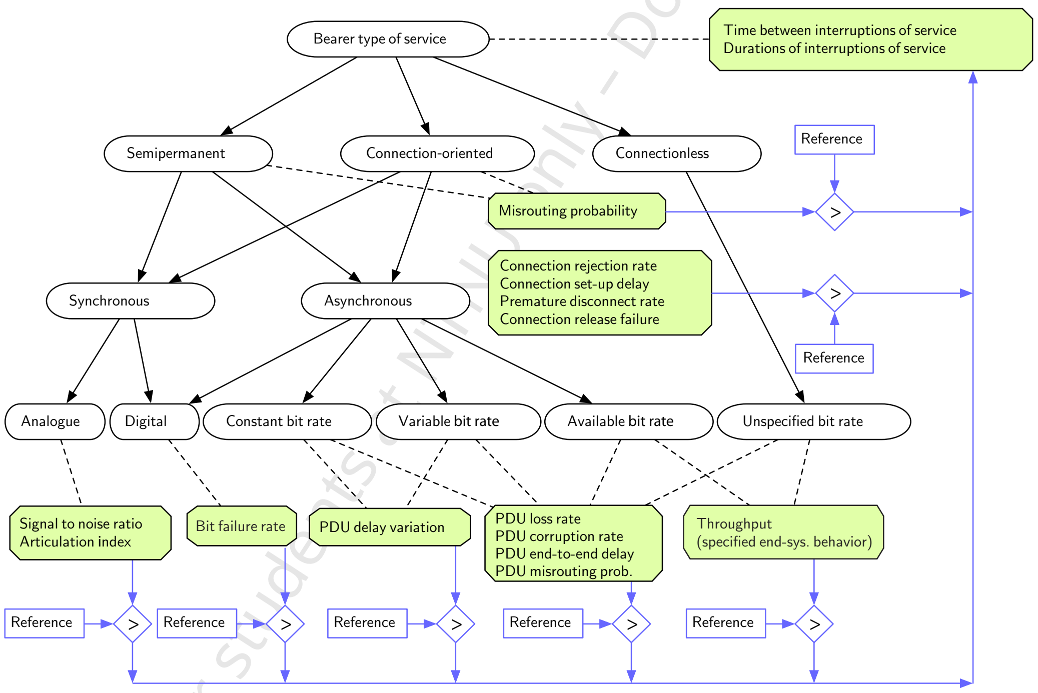


Figure 1.18: Illustration of how the performance parameters can be used to determine whether the system is working or not

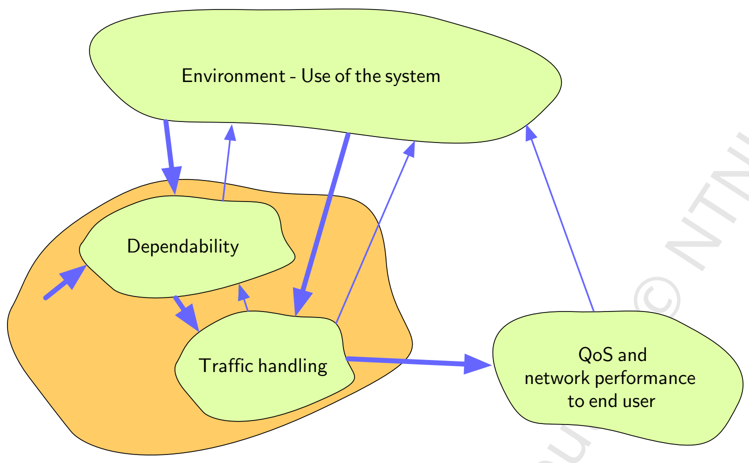


Figure 1.19: Relations between properties of a system

Figure 1.19 shows also that the environment of the system influences both:

- the behavior (dependability) of the system through
  - physical operating conditions,
  - operation and maintenance, and
  - offered traffic
- and the traffic handling (performance) through the offered traffic characterized by
  - the traffic,
  - the traffic interests, and
  - the traffic mix.

In addition to the main influences symbolized by the bold arrows in Figure 1.19, there are also retroactions, indicated by the thin arrows, that have to be considered when modeling, analyzing and dimensioning the system.

To sum up, the QoS of a system is determined by the occurrence of errors, the performance with and without errors, and the operational condition the system is exposed to. Note that from the user's point of view the distinction between loss and unavailability may be unclear.

1.2.5 QoS specification

Standardized guidelines have been developed for measuring and specifying the QoS for some services. A relevant way of doing this is the 3×3 matrix method from the ITU recommendation I.350 shown in Table 1.2.

Table 1.2: 3×3 **matrix approach for QoS specification for a connection-oriented service according to ITU-T recommendation I.350 [15]**

		Functions		
		Access	Information transfer	Disengagement
Service quality criteria	Speed	Connection set-up delay – Mean – Maximum – Quantile[a]	Propagation delay Jitter – Mean – Maximum – Quantile	Disconnection delay – Mean – Maximum – Quantile
	Accuracy	Connection set-up error Incorrect access (e.g. wrong destination) – Rate[b] – Ratio[c]	Information transfer error Severely errored period – Rate – Ratio	Premature disconnection Disconnection error – Rate – Ratio
	Dependability	Connection set-up denial – Rate – Ratio	Information loss “Random” information – Rate – Ratio	Disconnection denial Disconnection failure – Rate – Ratio

[a] Only a given proportion of events, e.g. 5%, exceed this value
[b] Events per time unit
[c] Proportion of the number of events, i.e. probability, see Section 2.1.1

Until now we have discussed only purely technical aspects of services and QoS. Service delivery includes additional aspects that matter for both the subscriber and the supplier, e.g. sales, administration and operation related to service delivery, accounting and billing. QoS parameters are also associated to these non-technical aspects. The matrix shown in Table 1.3 covers all the aspects related to the delivery of a commercial service and includes parts of Table 1.2. It has been elaborated by the Federation of Telecommunications Engineers in the European Commission (FITCE), and later adopted by the European Telecommunications Standards Institute (ETSI). See [13] for a definition of the matrix elements.

1.3 Use of modeling in development and dimensioning

This section gives insight into how the methods presented in the next chapters may be used in the planning, development, dimensioning and operation of ICT systems.

Table 1.3: Example of a generic framework to specify QoS requirements [13]

Service function		Service quality criteria						
		Speed	Accuracy	Availability	Reliability	Security	Simplicity	Flexibility
Sales								
Service management	Provision							
	Alteration							
	Service support							
	Repair							
	Cessation							
Technical quality	Connection establishment							
	User information transfer							
	Connection release							
Billing								
Service management by the user								

1.3.1 Requirements and specifications

We have introduced dependability and performance as non-functional properties of a system, and motivated their significance. Requirements on these properties must appear in the specification of an ICT-system. A system specification should consist of requirements on:

- *functional properties*, i.e. which functions are performed, the man-machine interface, etc.
- *non-functional properties* which include dependability and performance. Examples of dependability and performance attributes are given in Section 1.2. Note that the system environments should also be specified, e.g. the operational conditions, the load to be handled, etc. See Figures 1.2 and 1.19.

The system cost is substantially influenced by its dependability and performance. The system cost includes among others:

- the costs of development, production, retail and guarantee for a producer,
- the costs of acquisition and operation for a service provider (and in some cases the penalties for not fulfilling the service level agreement should be also considered, see page 22), and
- the costs of purchase, utilization and repair for an end-user of an equipment or a service.

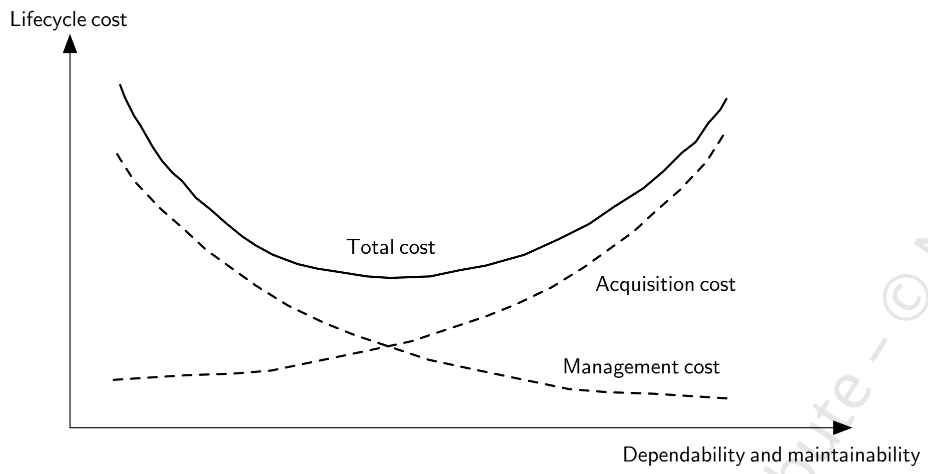


Figure 1.20: Life-cycle cost as a function of dependability and maintainability

How these parameters relate is complex and goes beyond the scope of this book. Figure 1.20 gives a sketch of the life-cycle costs of a system as a function of its dependability and maintainability properties. If no or little attention is given to these properties during the development of a system (poor quality assurance, no fault tolerance, cheap components, no maintenance support, etc.) the acquisition cost is low, but repair and maintenance expenses (in some systems the cost of a failure may be significant compared to the system cost) are high, so the life-cycle cost is also high. On the other hand, development costs rise rapidly when increasing the requirements on dependability and maintainability, so that the total cost is also high when the requirements are overestimated.

In addition to purely financial trade-offs, the system specification should also contain requirements on dependability and performance to ensure that the system will actually be usable. For example, a voice service will not be usable if the delay and/or the packet loss are too high. Besides, it must be noticed that both dependability and performance properties may, for some applications, have a marketing effect greater than their rational/financial significance.

1.3.2 Methods

Two methods for analysis and dimensioning of ICT systems are presented in this book, namely *mathematical analysis* and modeling based on the theory of stochastic processes and *simulation*. Another possible approach is to carry out *measurements* on a prototype of a system or on a real system. This section briefly lists pros and cons of each of these methods.

It is not a matter of choosing one of these methods. Both mathematical analysis and simulation may be performed to take benefit of both approaches, and it is even often required. Simulation is used to validate the simplifications and assumptions made in the mathematical model. Measurements on a prototype or on the first system produced should also always be carried out to ensure that the system fulfills its requirements.

Mathematical analysis

- Pros
  - In the case of a well-known problem, results are rapidly and easily obtained using formulas or algorithms.
  - Explicit parametric relations can be found for simple models, and parametric analysis, e.g. sensitivity analysis, is relatively easy.
- Cons
  - The level of knowledge required to design new models is relatively high, and models are generally not flexible in case of changes to the system.
  - Detailed models may become prohibitively complex and result in state spaces that are too large to be tractable. For dependability analysis, some models called *structural models*, see Section 7.3, mitigate this drawback by disregarding the system behavior.
  - Assumptions and simplifications, e.g. in terms of distributions, independence of observations, etc., must often be made and it may be hard to foresee the validity of the results.

Simulation

- Pros
  - Relatively detailed and realistic models can be used. Simulation allows to use system models with an arbitrary level of details. The challenge is to include all that is relevant for the evaluation but nothing more.
  - A lower level of knowledge is required to design new models.
- Cons
  - Computation times may be very long, in particular if the time constants have very different orders of magnitude.
  - Large stochastic uncertainties may be associated with the results and it may be hard to interpret the results correctly.
  - Parametric analysis is very demanding and it may be infeasible to derive explicit parametric relations even for simple systems.

Measurements on (a prototype of) a system

- Pros
  - Results are trustworthy since they are not affected by assumptions and simplifications.
  - Results are realistic and detailed.
- Cons
  - Measurements can only be performed after an existing prototype or system has been produced. Hence, measurements can not be used at early stages of the development process.
  - Realistic, stable, controllable and reproducible operation/measurement environment may be very hard and costly to obtain. Note however that despite this shortcoming, measurement results are often considered trustworthy.
  - The measurement of processes with large time constants may imply very long observation times and specific techniques may be required. For instance, measurements of dependability attributes on systems with long times to failure can only be carried out using fault injection techniques, or in the case of follow-up measurements on a system in operation.
  - Measurements suffer similar limitations as simulation with respect to stochastic uncertainty and parametric analysis.

1.3.3 Design-evaluation cycle

It is important to account for dependability and performance for all types of systems (components); end-user systems, network elements (e.g. switch or router), service platforms such as data processing and database services, service delivery, network administration,etc. This applies both to the development of a new system and to system planning and integration, where the dependability and performance properties of the system components must be controlled.

A common flaw in system planning and development is that dimensioning and evaluation of dependability and performance are not carried out as an integrated part of the work. This may be to save time and money (in the short term), or due to a lack of expertise, but in any case is likely to yield problems when:

- the system does not work as expected under real operational conditions (environment, load, etc.).
- the potential customers set requirements for the dependability and performance properties of the system and ask for documentation showing that the system does fulfill these requirements.
- the system is more expensive than necessary because of over- or incorrect dimensioning.

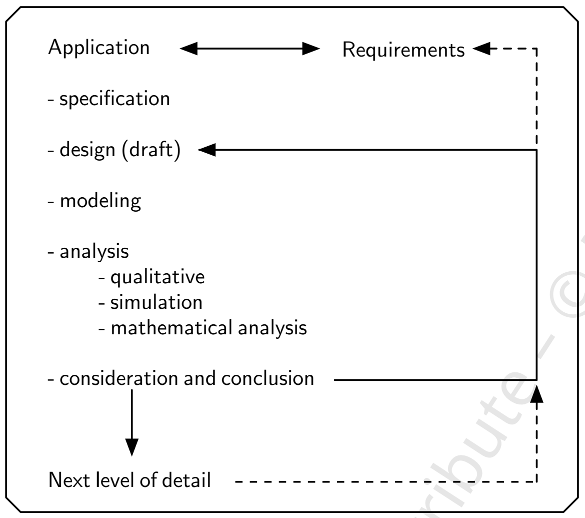


Figure 1.21: Design and evaluation procedure

The non-functional aspects of the system should be handled with the same attention as the functional aspects when specifying and modeling the system. Both types of properties must be specified before the system is developed/purchased and a thorough design and modeling must be done before the system is built.

The evaluation of dependability and performance should also be part of the planning or design work as shown in Figure 1.21.

In some cases, it is necessary to establish a dependability and performance budget. This budget shows how much the various system components are allowed to contribute to the lack of these properties in the system. For instance, for a component how much the different types of faults are allowed to impact the system availability, or how much the different network elements are allowed to contribute to the end-to-end delay and to the packet loss.

An important aspect is how well the system scales with respect to these properties. For instance:

- does the availability of the services delivered to the end-user decrease when the size of the system (number of requests and amount of resources to serve them) increase? This should not happen. (In risk evaluation one, on the contrary, tends to increase the dependability requirements when the consequences - here the number of simultaneously affected users - increase.)
- does the end-to-end delay stay within the specified bounds when the size of the system and when the load on are scaled up to the values the system has been dimensioned for?

By making dependability and performance models for a system, one is forced to thoroughly reason about the system design with another approach than when focusing only on purely functional aspects. This gives a better comprehension of how the system works and contributes to identify the possible weaknesses of the functional aspects of the system design.

2 Fundamentals of probability

Imagine yourself wandering downtown. You have read in the newspaper that one in ten call attempts is blocked. You may think: I called at least ten times during the last week and none of my calls was lost. Does it mean that what is written in the newspaper is wrong? On the other hand you remember well that, once, four of your attempts were blocked in a row in less than five minutes! Maybe has the network been improved lately? Or maybe have you been lucky last week?

Most people may think that one in ten call attempts blocked means that on average every user sees one in ten of his/her attempts blocked. Some may believe that every single person gets 10% loss, while others may assume that it depends on when and from where people is calling. A 10% loss on average can mean that ten users get 10% each, but also that one of them gets all his/her call attempts lost (e.g. because the base station in the region he/she is calling from is down) and the other nine never get lost. The overall average value alone does not allow to say anything about the distribution of the call attempts loss.

As we will see, we must assume that the operation of ICT-systems, e.g. a mobile phone network, can be described using a stochastic model that can vary over time. This model has, among other things, to capture the fact that a loss can be caused by a system outage as well as by a system overload due to an excessive number of users calling simultaneously.

This chapter provides details on how this stochastic operation can be described by means of distribution functions, how different characteristics such as mean values and variations can be calculated, and how dependencies between stochastic processes affect the results. Supplementary literature includes the general textbooks [1, 2] and the applied books [16, 17]. In Chapters 6 and 7 we will see in more details which stochastic processes are relevant to describe various human and physical phenomena in ICT-systems.

2.1 Probability calculation

2.1.1 Probability

If we carry out experiments with a deterministic process, the outcome will be identical as long as the initial conditions remain the same. For instance if the blocking process of our mobile phone system was deterministic and every tenth call was blocked (for some mysterious reason), then we would know that after observing 200 call attempts exactly 20 would have been blocked.

On the other hand, if we assume that the blocking process is governed by some randomness, we must use stochastic processes and probability models to describe it. Examples of such randomness or stochastic behavior include: the number of call attempts within a coverage zone varies from day to day, each successful call results in a conversation of variable duration, radio transmissions are prone to interferences, the base station may become unavailable. If we count the number of successful and blocked calls in the same area keeping the same number of potential, not necessarily active, users, the result will not be 10% each time. However, if it is known that there is "a 10% probability that a call attempt is blocked under given conditions", it means that if we perform a large number of experiments under the same conditions then we will get a result close to 10%.

Experiment

An experiment is any process or study whose outcome is random and that results in an observation. The set of all possible outcomes of an experiment is called the *sample space* and is denoted Ω.

Event

An event is any outcome, or collection of outcomes, of an experiment. It is a subset of the sample space.

This can be represented by means of Venn diagrams, see Figure 2.1. We say that the event A has *occurred* when the outcome of an experiment belongs to the subset A of Ω, $A \subseteq \Omega$. The *complement* $\bar{A}$ of the event A is the subset of all elements of Ω that are not in A. In our example, an event is either a successful or a blocked call attempt.

The *relative frequency* of a particular event is the number of times this event has occurred divided by the number of times an experiment has been carried out.

Probability

The *probability* of an event is defined as its asymptotic relative frequency, i.e. when the number of experiments tends to infinity.

In practice, the number of experiment is chosen large enough to achieve a given level of accuracy. This latter point will be discussed more formally in Chapter 8.

Let:

Ω	sample space
N_Ω	number of events in (*cardinality* of) Ω
A	set of events of interest
N_A	number of outcomes A observed

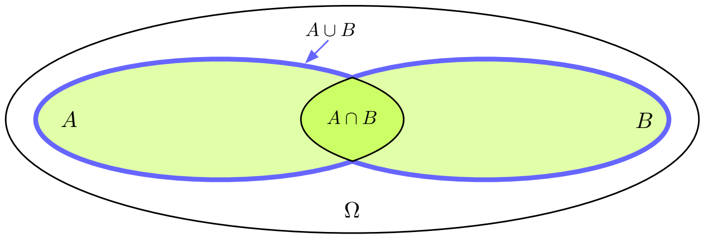


Figure 2.1: Venn-diagram: union and intersection of events A and B

The probability that event A occurs, $P(A)$, is:

$$P(A) = \frac{N_A}{N_\Omega} \tag{2.1}$$

It follows directly that the probability of any event A in Ω is greater or equal to 0 and smaller or equal to 1.

$$0 \leqslant P(A) \leqslant 1, \ \forall A \in \Omega \tag{2.2}$$

When performing experiments, the sample space and the events must be carefully defined. For example if we want to determine the asymptotic unavailability of a system, i.e. the probability that the system is in a failed state, we can define the state changes as possible events and hence "the system fails" as the event of interest. But we will get a relevant value for the system unavailability only under very specific assumptions. On the other hand, we can observe the system at either random or fixed points in time, observations times being independent of the system operation. Each observation is then an event, while observing the system in a failed state is an event of interest. The number of events of interest divided by the number of observations gives the unavailability of the system.

Example 2.1 Call congestion
During an experiment, the number of call attempts observed within the coverage area of the MobileCom mobile phone network is $N_\Omega = 100000$. Among these, $N_A = 1500$ calls have been lost. The set of possible events is Ω="call attempt" and the set of events of interest A="call lost". Using (2.1), we find the blocking probability $P(A) = N_A/N_\Omega = 1.50\%$.

Example 2.2 Unavailability of a base station
During two years, MobileCom gathered monitoring data for its 10 base stations. Each base station state has been observed once every hour during these two years. The set of observations is Ω="base station is observed" and the set of events of interest A="base station is observed in a failed state". The cardinality of Ω is $N_\Omega = 24 \cdot 365 \cdot 2 \cdot 10$, while the monitoring data show that $N_A = 1400$. Using (2.1), we find the probability that a base station is unavailable $U = P(A) = N_A/N_\Omega = 0.80\%$. In addition, monitoring data show the total downtime for all these 10 base stations is 84571 minutes during the two years of operation. The unavailability is equal to the expected time the system is in a failed state, see Chapter 1, i.e. $U = 84751/(60 \cdot 24 \cdot 365 \cdot 2 \cdot 10) = 0.81\%$.

Note that in both cases it is assumed that base station failures are independent. Both approaches give the same result if the observation period is sufficiently long and the number of samples sufficiently large.

2.1.2 Intersection and union of events

If the sample space Ω includes an event of interest B in addition to A, we can calculate the *joint probability* $P(A \cap B)$, i.e. the probability that both events occurs in conjunction (*intersection*), and the probability $P(A \cup B)$ that one or both of these events occur (*union*), see Figure 2.1.

Additive rule

$$P(A \cup B) = P(A) + P(B) - P(A \cap B) \tag{2.3}$$

Mutual exclusivity

Two events A and B are *mutually exclusive* if and only if:

$$P(A \cap B) = 0,\ P(A) \neq 0 \text{ and } P(B) \neq 0 \tag{2.4}$$

In the general case, the state space includes an arbitrary number of events.

Partition

A set of events $\{A_i\}$ forms a *partition* of the sample space Ω if and only if:

$$\begin{array}{ll} 1. & P(A_i \cap A_j) = 0,\ \forall i \neq j \ \ \textit{(mutual exclusivity)} \\ 2. & \sum_{\forall i} P(A_i) = 1 \ \ (\{A_i\} \textit{ covers } \Omega) \end{array} \tag{2.5}$$

2.1.3 Conditional probability

In some situations, the probability of an event depends on the occurrence of other events. For example A="call lost" depends on B="base station is down". The *conditional probability* is the probability of some event, given the occurrence of some other event.

Conditional probability

If A and B are two events, and $P(B) > 0$, then the conditional probability of A given B is:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)} \tag{2.6}$$

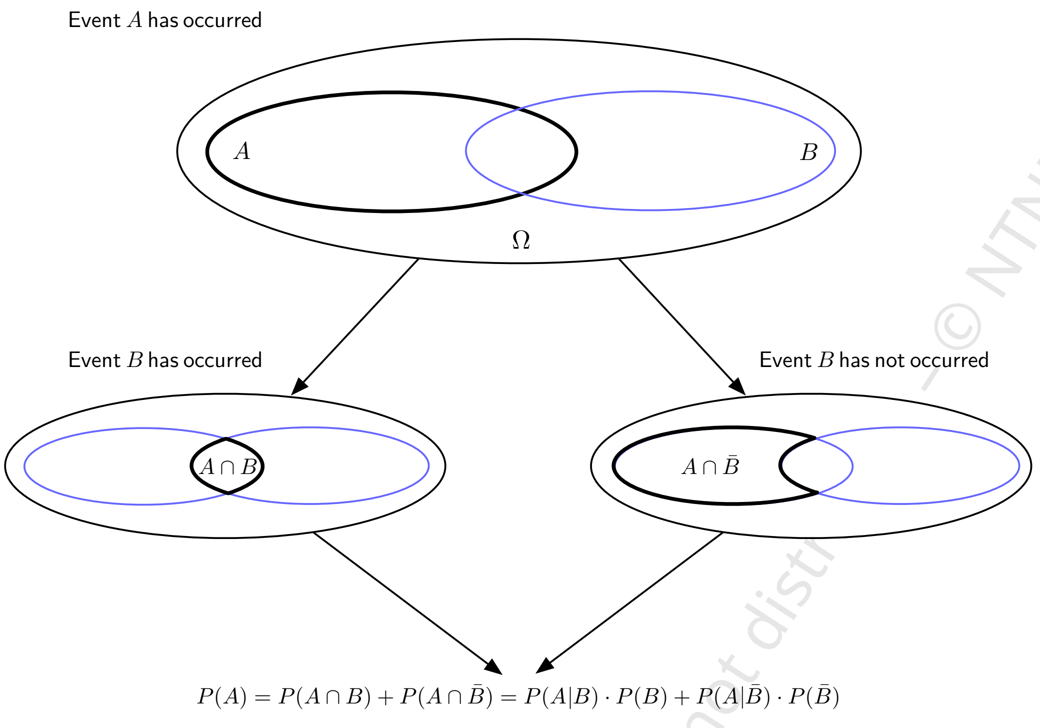

Event A has occurred
A
B
Ω
Event B has occurred
Event B has not occurred
$A \cap B$
$A \cap \bar{B}$
$P(A) = P(A \cap B) + P(A \cap \bar{B}) = P(A|B) \cdot P(B) + P(A|\bar{B}) \cdot P(\bar{B})$

Figure 2.2: Total probability

Equation (2.6) can also be written:

Multiplicative rule

$$P(A \cap B) = P(A \mid B) \cdot P(B) \tag{2.7}$$

Law of total probability

There may be a limited number of events B that affect the probability of an event A, see for example Figure 2.2. The *law of total probability* gives the *marginal probability* of event A, $P(A)$, using conditional probabilities.

$$P(A) = P(A \cap B) + P(A \cap \bar{B}) = P(A \mid B) \cdot P(B) + P(A \mid \bar{B}) \cdot P(\bar{B}) \tag{2.8}$$

More generally,

Law of total probability

When $\{A_i\}$ forms a partition of the sample space, then for any event B:

$$P(B) = \sum_j P(B \cap A_j) = \sum_j P(B \mid A_j) \cdot P(A_j) \tag{2.9}$$

Bayes' formula

Bayes' formula relates the conditional and marginal probabilities of events A and B.

$$P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B)} \tag{2.10}$$

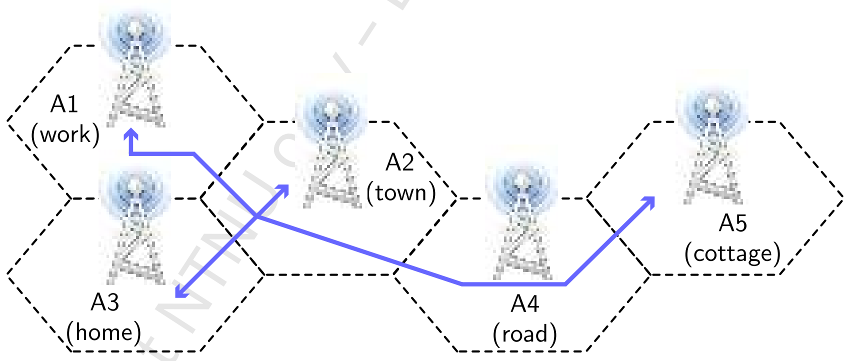
More generally,

Bayes' formula

When $\{A_i\}$ forms a partition of the sample space [2, 1]:

$$P(A_i \mid B) = \frac{P(B \mid A_i) \cdot P(A_i)}{\sum_j P(B \mid A_j) \cdot P(A_j)}, \forall i \tag{2.11}$$

Example 2.3 Mr Jensen on the move - How often are his calls lost?
Mr Jensen drives from home to work through the city every day and spends his spare time either in the city or at his cottage. He always keeps his mobile phone with him and uses it equally often regardless the place where he is. His travel pattern is shown in the figure below. The area considered is divided in 5 cells which correspond to the coverage zones of 5 base stations.



Mr Jensen's travel pattern has been observed during a large number of weeks to obtain precise estimates of the relative times he spends in the different cells. He is $t_1 = 0.2381$ of the time in cell A_1, $t_2 = 0.1250$ in cell A_2, $t_3 = 0.4672$ in cell A_3, $t_4 = 0.0268$ in cell A_4, and $t_5 = 0.1429$ in cell A_5. The probability $P(A_i) = t_i$ that Mr Jensen is in cell $i = 1, \ldots, 5$ is assumed to be equal to relative sojourn time in cell i. Besides, the mean congestion probabilities in each cell $P(B \mid A_i)$ are known from the mobile phone operator's data: $P(B \mid A_1) = 0.15$, $P(B \mid A_2) = 0.25$, $P(B \mid A_3) = 0.05$, $P(B \mid A_4) = 0.45$ and $P(B \mid A_5) = 0.15$. If there is congestion, any call to or from the cell is lost.

The probability that Mr Jensen's call attempt is lost assuming that it cannot be lost at the callee's end is obtained using the law of total probability (2.9):

$$P(B) = \sum_{j=1}^{5} P(B \mid A_j) \cdot P(A_j) = 0.15 \cdot 0.2381 + 0.25 \cdot 0.1250 + \ldots + 0.15 \cdot 0.1429 = 0.1238$$

The probability that Mr Jensen is in the city when he gets a call lost is obtained using Bayes' formula (2.11):

$$P(A_2 \mid B) = \frac{P(B \mid A_2) \cdot P(A_2)}{P(B)} = \frac{0.25 \cdot 0.1250}{0.1238} = 0.2524$$

2.1.4 Statistical independence and dependence

Statistical independence

Statistical independence

Two events A and B are *statistically independent* if and only if [2]:

$$P(A \mid B) = P(A) \Leftrightarrow P(B \mid A) = P(B) \Leftrightarrow P(A \cap B) = P(A) \cdot P(B) \qquad (2.12)$$

In other words, the probability that the event A occurs is the same regardless of the occurrence of the event B and vice versa.

More generally,

Mutual independence

A set of events $\{A_i\}$ are *mutually independent* if and only if for any finite subset $A_1, \ldots, A_n$

$$P(A_1 \cap \ldots \cap A_n) = P(A_1) \cdot \ldots \cdot P(A_n) \qquad (2.13)$$

Note that mutually independent events cannot be mutually exclusive and conversely.

Statistical dependence

Two events A and B are *statistically dependent* if the probability that A occurs depends on the occurrence of B.

We distinguish between:

- Cross dependence: The events A and B follow two different probability density functions, f_A and f_B respectively, see Section 2.2. For example, B="subscriber B is blocked" will depend on A="subscriber A calls".

- Time dependence: The events A and B follow the same probability density functions, $f_A = f_B$, but occurs apart in time. For example, the probability that B="loss at time t" may depend on A="lost call at time $t - 30s$".

Dependences are quantified using the *covariance* and the *correlation coefficient*, see Section 2.3.6. Time dependence is further detailed in Chapter 3.

2.2 Distribution of random variables

In real life it is hard to find a situation where everything is predictable. Usually there are one or more elements that make the situation unpredictable and its outcome random. There is a variety of causes of randomness for ICT-systems. For instance, the call duration varies from call to call and users neither always call the same number, nor at the same time, nor from the same place.

To construct a model of the reality, one needs to study the different sources of randomness and identify possible regularities that can be described using stochastic models. Experiments are carried out to provide a picture of the distribution of the process outcome. The outcome is a *random variable* and its distribution is called a *probability distribution*. A random variable may be uni- or multi-variate and we speak then of a *random vector*.

2.2.1 Random variable

Continuous random variable

A *continuous random variable* can take an infinite number of possible values. The sample space is constituted by one or several intervals. Examples include time distributions, e.g. call duration, time to failure, repair delay time.

Discrete random variable

A random variable is *discrete* if the sample space is finite or infinite but countable[1]. In this book, discrete variables will often be integers, e.g. the number of call attempts before success, the number of requests per second received by a web-server.

[1]A series of number $1, 2, 3, \ldots, \infty$ is infinite but countable, while a continuous series of points on the time axis between 0 and 1 is infinite and uncountable.

Notations

Random variables are represented using capital letters, while lower-case letters are used to represent the values they can take.

Let:

X	random variable
x_i	value of X for the i^{th} trial

In the context of ICT-systems, T is preferably used to represent a time, W a waiting time and N a number. When modeling large systems, subscripts are commonly used to differentiate between random variables, e.g. T_{CPU}, T_{start}. Capital letters are, however, not exclusively used to represent random variables. Other variables are also traditionally represented using capital letters.

Specific notations have been defined in the traffic and dependability fields and do partially use the same letters. It may be confusing since a letter will have a different meaning depending on the context. This is something one must be aware of and which incites to define clearly and explicitly the symbols used.

2.2.2 Probability density function and cumulative distribution function

Continuous random variable

Cumulative distribution function (continuous)

The *cumulative distribution function* (abbreviated *cdf*), or simply *distribution*, of a continuous random variable X is a non-decreasing function defined as:

$$F_X(x) = P(X \leqslant x),\ \forall x \in \Omega \tag{2.14}$$

where Ω denotes the sample space.

By definition,

$$0 \leqslant F_X(x) \leqslant 1,\ \forall x \in \Omega \tag{2.15}$$

Probability density function

The *probability density function (pdf)* of a continuous random variable X is:

$$f_X(x) = \lim_{\mathrm{d}x \to 0} \frac{P(x < X \leqslant x + \mathrm{d}x)}{\mathrm{d}x} = \lim_{\mathrm{d}x \to 0} \frac{F_X(x + \mathrm{d}x) - F_X(x)}{\mathrm{d}x},\ \forall x \in \Omega \tag{2.16}$$

When $F(x)$ is differentiable over Ω

$$f_X(x) = \frac{\mathrm{d}F_X(x)}{\mathrm{d}x} \tag{2.17}$$

A probability density function $f_X(x)$ satisfies:

$$\begin{aligned} &1. \quad f_X(x) \geqslant 0,\ \forall x \in \Omega \\ &2. \quad \int_{x\in\Omega} f_X(x)\,\mathrm{d}x = 1 \end{aligned} \tag{2.18}$$

When several random variables are used, as it is often the case, it is appropriate to use the variable as an index as it is done here. However $F(x)$ and $f(x)$ may also simply be used when there is no ambiguity with respect to the variable they refer to.

Discrete random variable

The cumulative distribution function of a discrete random variable X is thus written:

Cumulative distribution function (discrete)

$$F_X(x) = P(X \leqslant x),\ \forall x \in \Omega \tag{2.19}$$

where Ω denotes the sample space.

Probability mass function

The *probability mass function (pmf)* gives the probability that a discrete random variable X is exactly equal to some value (point-probability).

$$f_X(x) = P(X = x),\ \forall x \in \Omega \tag{2.20}$$

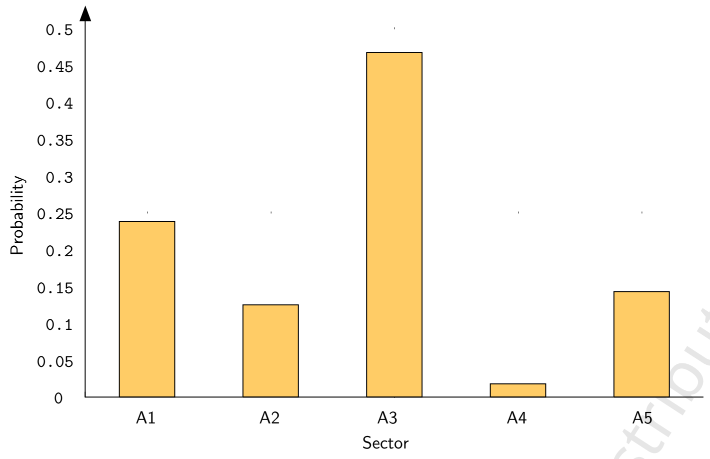
When the random variable needs to be explicitly mentioned, $P(X = x)$ is used.

A probability mass function $f_X(x)$ fulfills:

$$\begin{aligned} &1. \quad f_X(x) \geqslant 0,\ \forall x \in \Omega \\ &2. \quad \sum_{x\in\Omega} f_X(x) = 1 \end{aligned} \tag{2.21}$$

Example 2.4 Mr Jensen on the move - Probability distribution
Mr Jensen's travel pattern model presented in Example 2.3 is a discrete distribution since the sample space is finite; the 5 cells (coverage zones). The probability that Jensen is in a given area A_i, p_{A_i},

is equal to the expected portion of time he spends in the area A_i. Remember that the sum of the probabilities has to be equal to 1.



2.2.3 Probability distributions used in the description of ICT-systems

The probability distribution presented in Example 2.4 is an example of an *empirical distribution*. The observed values are normalized so that the sum is 1 (otherwise it is not a probability distribution) and plotted over the sample space. An empirical distribution has the advantage of including everything that has been observed, but on the other hand this becomes a disadvantage when the amount of data handled is large. For instance, recording all the IP-traffic on a local area network during a whole day may result in a very large amount of data. It is preferable to find a parametric model which describes the studied process as close to the reality as possible. Thus, it is not necessary to record all the data. Instead it is sufficient to record only the information needed to compute the parameters of interest. Empirical distributions are used when no known parametrized probability distribution, or combination of probability distributions, fits.

A parametric model enables to perform analytical calculations to determine the characteristics of the studied system. A simulation model in which values are drawn randomly according to the parametric distribution may also be used for the same purpose, see Chapter 4.

A parametric model of a random variable is a function with one or more parameters. $X \sim \mathcal{F}(\lambda)$ is used to express that the random variable X follows a distribution $\mathcal{F}$ with parameter λ. In the rest of the section, examples of relevant parametric distributions used in teletraffic and dependability analysis are given.

Continuous distributions

Negative exponential distribution (n.e.d.) The probability density function of the *negative exponential distribution*, or simply *exponential distribution*[2] , is:

$$f(x) = \lambda e^{-\lambda x},\ x \in \mathbb{R}^+ \tag{2.22}$$

where $\lambda \in \mathbb{R}^*_+$ is the rate parameter.

The cumulative distribution function is:

$$F(x) = 1 - e^{-\lambda x},\ x \in \mathbb{R}^+ \tag{2.23}$$

The negative exponential distribution is the most frequently used distribution in teletraffic and dependability modeling. There are many reasons for this, but the two most important are that it has properties which simplify the solution of complex analytical models and that it captures the behavior of real-world stochastic processes. It is often used to model the time between events that occur at a constant intensity. For instance, it can be used as a good approximation of the distribution of the time between two phone calls in a telephone exchange.

If the studied process cannot be modeled by a negative exponential distribution, new distributions can easily be built by combining several negative exponential distributions. Such distributions are called *phase-type distributions*.

Hypo-exponential distribution The *hypo-exponential distribution* is the distribution of the sum (series connection) of k independent exponentially distributed random variables that may have different rates.

Erlang-k distribution The *Erlang-k distribution*, or simply *Erlang distribution*, is a special case of the hypo-exponential distribution where the k random variables are identically distributed. It is defined by two parameters: the shape, $k \in \mathbb{N}^*$, and the rate of each phase, $\lambda \in \mathbb{R}^*_+$.

$$X \sim \text{Erlang-}k(\lambda) \Leftrightarrow X = \sum_{i=1}^{k} Y_i,\ Y_i \sim \text{n.e.d.}(\lambda) \tag{2.24}$$

The probability density function of the Erlang distribution is:

$$f(x) = \frac{(\lambda x)^{k-1}}{(k-1)!}\lambda e^{-\lambda x},\ x \in \mathbb{R}^+ \tag{2.25}$$

[2]The distribution is called *negative* because of the minus in front of the parameter λ ($\lambda > 0$). This gives a decreasing density function defined over $\mathbb{R}^+$. Unless otherwise stated, exponential distribution refers to the negative exponential distribution in this book.

and the cumulative distribution function:

$$F(x) = 1 - e^{-\lambda x} \sum_{j=0}^{k-1} \frac{(\lambda x)^j}{j!},\ x \in \mathbb{R}^+ \tag{2.26}$$

When k equals 1, the Erlang-k distribution simplifies to an exponential distribution.

Example 2.5 Time between call attempts
If the time between two call attempts is n.e.d., then the time between every tenth call attempt is Erlang-10 distributed.

Example 2.6 Stock control
It is known by experience that the lifetime of a media roll in a PrintMe2000 printer is negatively exponentially distributed. The operating department, which has 20 printers of this type, has a stock with room for 12 media rolls. They have the strategy to order 10 new media rolls when there are only two left in stock. Therefore the time between each order is the sum of 10 independent and identically n.e.d. random variables, i.e. the time is Erlang-10 distributed.

Gamma distribution The *Gamma distribution* is a generalization of the Erlang distribution where the shape parameter k is not restricted to integer values, $k \in \mathbb{R}^*_+$, and $\lambda \in \mathbb{R}^*_+$ is denoted the scale parameter. The probability density function becomes:

$$f(x) = \frac{(\lambda x)^{k-1}}{\Gamma(k)} \lambda e^{-\lambda x},\ x \in \mathbb{R}^+ \tag{2.27}$$

and the cumulative distribution function:

$$F(x) = \frac{1}{\Gamma(k)} \int_0^{\lambda x} u^{k-1} e^{-u}\, \mathrm{d}u,\ x \in \mathbb{R}^+ \tag{2.28}$$

where $\Gamma(k) = \displaystyle\int_0^{\infty} e^{-x} x^{k-1}\, \mathrm{d}x,\ k \in \mathbb{C}$ and $\mathrm{Re}(k) > 0$ is the Gamma function[3].

Hyper-exponential distribution The *hyper-exponential distribution* is the weighted sum (parallel connection) of $k \in \mathbb{N}^*$ exponential distributions, each having a different rate parameter $\lambda_k \in \mathbb{R}^*_+$. The cumulative distribution function of the hyper-exponential distribution is given by:

$$F(x) = 1 - \sum_{j=0}^{k-1} a_j e^{-\lambda_j x},\ x \in \mathbb{R}^+ \tag{2.29}$$

where $\sum_{j=0}^{k-1} a_j = 1$.

[3]When $k \in \mathbb{N}^+$, $\Gamma(k) = (k-1)!$.

Weibull distribution The probability density function of the *Weibull distribution* is:

$$f(x) = k\lambda(x\lambda)^{(k-1)}e^{-(x\lambda)^k},\ x \in \mathbb{R}^+ \tag{2.30}$$

where $k \in \mathbb{R}_+^*$ is the shape parameter and $\lambda \in \mathbb{R}_+^*$ the scale parameter of the distribution. The cumulative distribution function is:

$$F(x) = 1 - e^{-(x\lambda)^k},\ x \in \mathbb{R}^+ \tag{2.31}$$

The exponential distribution is a special case of the Weibull distribution ($k = 1$). The Weibull distribution is often used in dependability modeling to model the lifetime of physical components.

Uniform continuous distribution The *uniform continuous distribution* is a parametric distribution which probability density function is constant over the sample space. It is parametrized by the smallest and largest values that the uniformly-distributed random variable can take, a and b. The probability density function of the uniform distribution is thus:

$$f(x) = \begin{cases} \frac{1}{b-a} & x \in [a,b] \\ 0 & x \notin [a,b] \end{cases} \tag{2.32}$$

and the cumulative distribution function:

$$F(x) = \begin{cases} 0 & x < a \\ \dfrac{x-a}{b-a} & x \in [a,b] \\ 1 & x \geqslant b \end{cases} \tag{2.33}$$

The uniform distribution is useful when nothing is known about a process except the smallest and/or largest possible values. Simple processes such as disk-access times are uniformly distributed.

All the distributions mentioned in this section are summarized in Table 2.1. Moreover, Figure 2.3 shows the shape of different probability density functions for a given set of parameters chosen such as the mean value is the same (12) in all cases. The figure also shows how these distributions relate to each other. For instance, take as a starting point a negatively exponentially distributed random variable X. The sum of k independent identically negatively exponentially distributed X's is Erlang-k distributed. The Erlang distribution is a special case of the Gamma distribution where k is an integer. According to the central limit theorem (see Section 8.2.2), the sum of a very large number of X's will approach a Normal-distribution. Finally, the k^{th}-root of $X \sim$ n.e.d.(λ) is Weibull-distributed with shape k and scale λ.

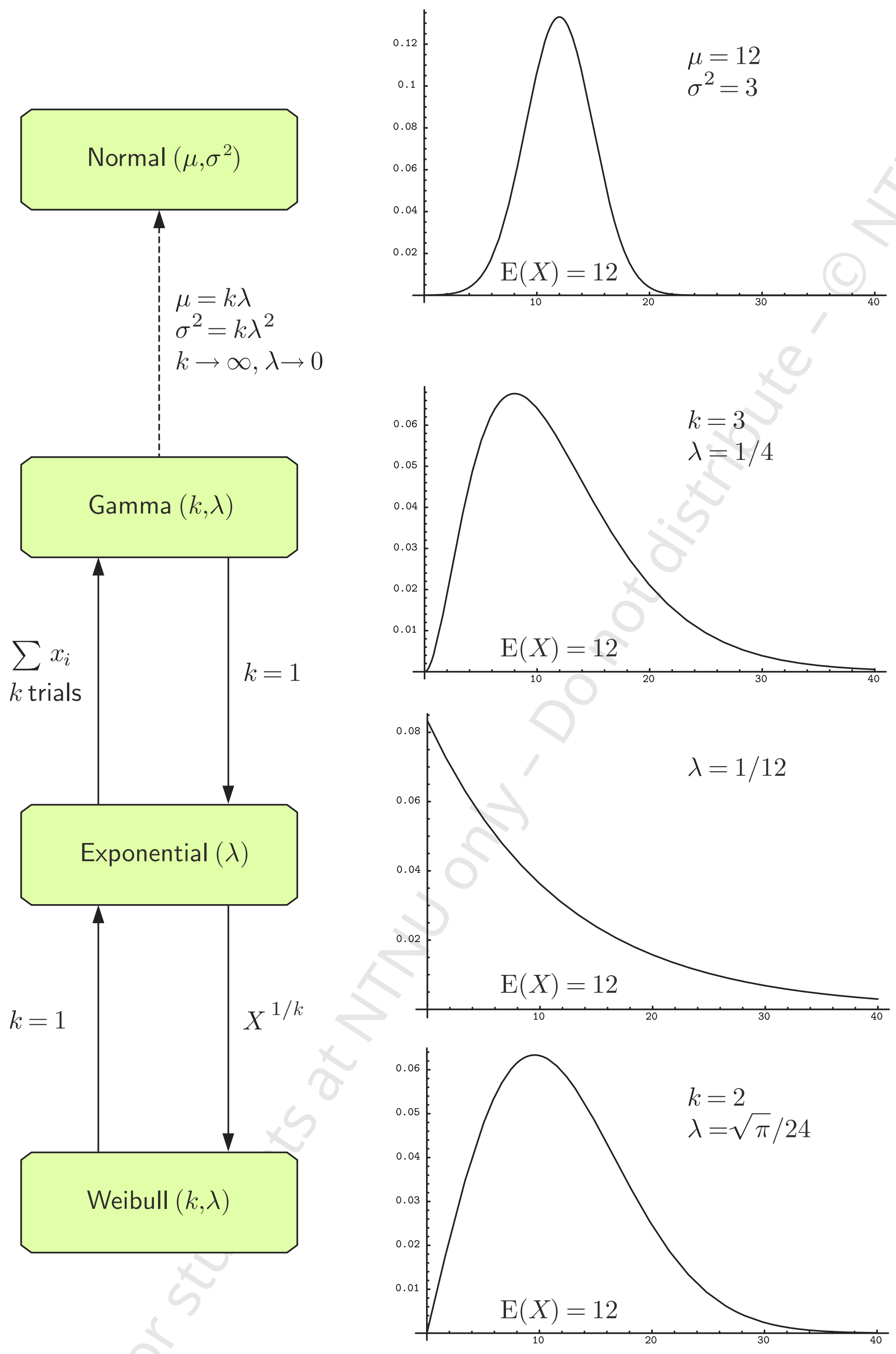


Figure 2.3: Relations between continuous distributions

Discrete distributions

Bernoulli distribution In the simplest case, there are only two possible outcomes, 1 or 0, which may be yes or no, success or failure, accepted or blocked etc. This is the *Bernoulli distribution* where the probability of outcome 1 is p and the probability of outcome 0 is $1 - p$ (remember that the sum of all the probabilities is 1).

$$f(x) = \begin{cases} p & x = 1 \\ 1-p & x = 0 \end{cases} \tag{2.34}$$

Geometric distribution The *geometric distribution* is the probability distribution of the number X of events 0 in a sequence of independent Bernoulli trials before the event 1 is observed. The probability mass function of the geometric distribution is the probability that the event 1 occurs on the $(x+1)^{\text{th}}$ trial.

$$f(x) = (1-p)^x p,\ x \in \mathbb{N} \tag{2.35}$$

$$F(x) = 1 - (1-p)^{x+1},\ x \in \mathbb{N} \tag{2.36}$$

For instance, X may model the number of successful calls made by Mr Jensen, see Example 2.3, before he is blocked.

The probability mass function of the *shifted (to 0)* geometric distribution is the probability that the event 1 occurs on the x^{th} trial.

$$f(x) = (1-p)^{x-1} p,\ x \in \mathbb{N}^* \tag{2.37}$$

Binomial distribution The *binomial distribution* is the probability distribution of the number X of events 1 in a sequence of n independents Bernoulli trials, $n \in \mathbb{N}^*$.

$$f(x) = \binom{n}{x} p^x (1-p)^{n-x},\ x \in \{0, 1, \ldots\} \tag{2.38}$$

For example, X may model the number of times Mr Jensen's calls, see Example 2.3, have been blocked over his last 100 attempts.

Poisson distribution The *Poisson*[4] *distribution* corresponds to the binomial distribution with an infinite number of Bernoulli trials. It expresses the probability of the number of occurrences of an event, given that it occurs on average at the intensity $\alpha \in \mathbb{R}^+$.

$$f(x) = \frac{\alpha^x}{x!} e^{-\alpha},\ x \in \mathbb{N} \tag{2.39}$$

All the distributions mentioned in this section are also summarized in Table 2.1. Figure 2.4 shows the relations between these distributions; in all cases $p = 0.4$.

[4]Simon-Denis Poisson (1781-1840). French mathematician, geometer, and physicist.

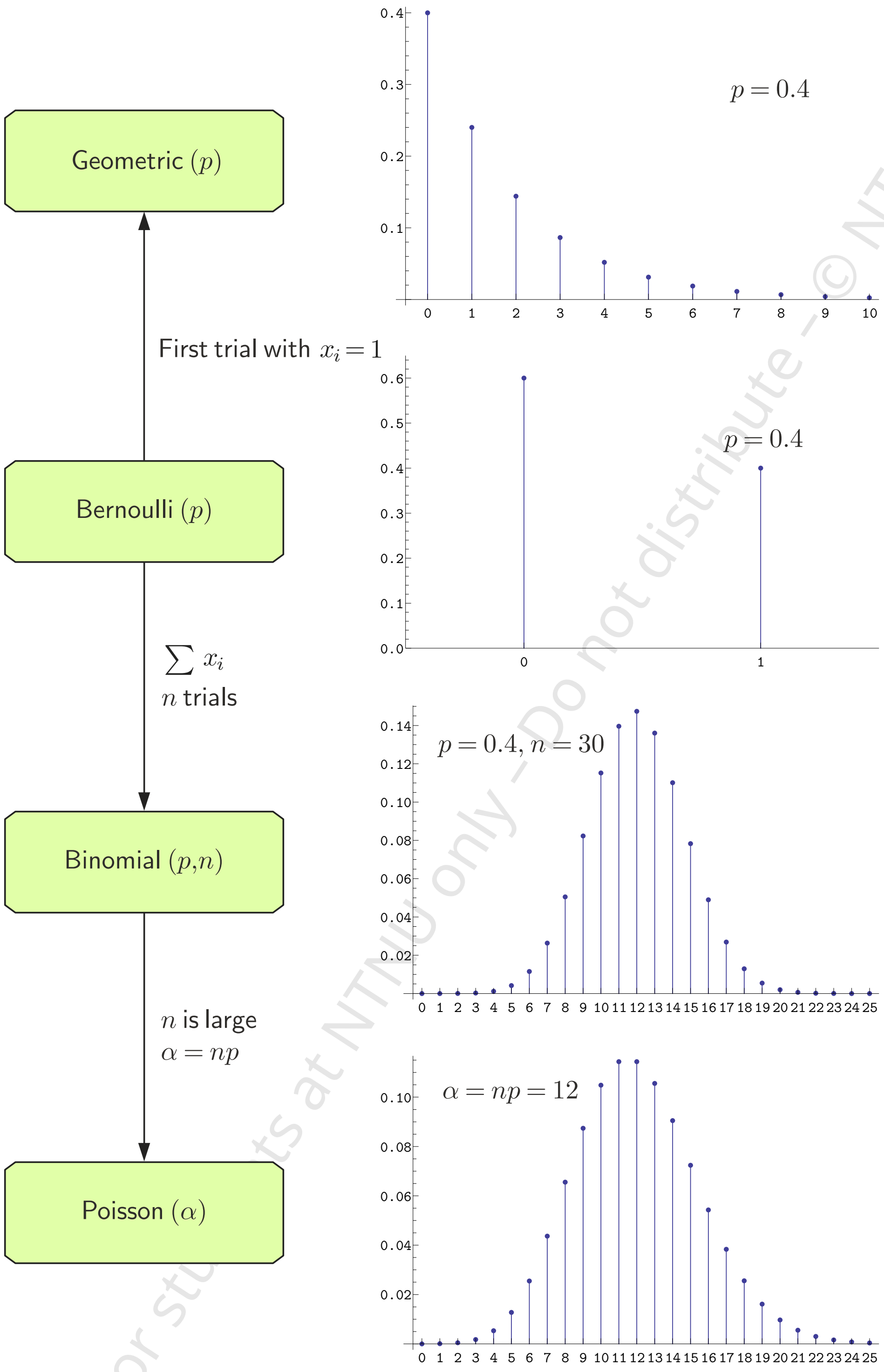


Figure 2.4: Relations between discrete distributions

2.2.4 Multiple random variables

So far we have only considered univariate random variables. In many occasions when analyzing the performance and the dependability of ICT-systems, univariate random variables are not enough and multiple random variables, or multivariate random variables, need to be regarded simultaneously. For instance, it may be relevant to study not only the size of packets sent over a network, but also the size of the packets and the source and destination of the packets; not only the duration of a flow, but also the duration of a flow and the type of traffic; not only the response time, but also the response time and the QoS class.

In the following, for the sake of simplicity, we consider simply a bi-variate random vector (X, Y) where X and Y denote two random variables defined on Ω_X and Ω_Y, respectively. However, the definitions can be generalized to a random vector of any size.

Joint probability distribution

The probability of the event (X, Y) is defined in terms of the probabilities of the corresponding events X and Y. The probability distribution of the simultaneous occurrence of X and Y is referred to as the *joint probability distribution*.

Joint probability distribution

- In the discrete case, the *joint probability mass function* is defined as:

$$f_{X,Y}(x,y) = P(X = x, Y = y) \tag{2.40}$$

  and for any region A in the xy plane

$$P((X,Y) \in A) = \sum_{(x,y) \in A}\sum f_{X,Y}(x,y) \tag{2.41}$$

- In the continuous case, $f_{X,Y}(x,y)$ is called the *joint probability density function* and for any region A in the xy plane

$$P((X,Y) \in A) = \iint_{(x,y)\in A} f_{X,Y}(x,y)\,\mathrm{d}x\,\mathrm{d}y \tag{2.42}$$

A joint distribution $f_{X,Y}(x,y)$, as any distribution, satisfies:

1. $f_{X,Y}(x,y) \geqslant 0, \ \forall (x,y)$ (2.43)
2. $\sum_{\Omega_X}\sum_{\Omega_Y} f_{X,Y}(x,y) = 1$ (discrete), $\int_{\Omega_X}\int_{\Omega_Y} f_{X,Y}(x,y)\,\mathrm{d}x\,\mathrm{d}y = 1$ (continuous)

Marginal probability distribution

Given the joint probability distribution $f_{X,Y}(x,y)$ of (X,Y), the probability distribution of X alone is obtained by summing/integrating $f_{X,Y}(x,y)$ over all the values of Y. This distribution is called the *marginal probability distribution* of X.

Marginal probability distribution

The *marginal probability distribution* of X is

$$f_X(x) = \sum_{\Omega_Y} f_{X,Y}(x,y) \text{ (discrete)} \quad \text{and} \quad f_X(x) = \int_{\Omega_Y} f_{X,Y}(x,y)\,\mathrm{d}y \text{ (continuous)} \tag{2.44}$$

Note that we cannot generally determine the joint distribution of (X,Y) if we only know the marginal distributions of X and Y.

Conditional probability distribution

Conditional probability distribution

The *conditional probability distribution* of Y given that $X = x$ is

$$f_{Y|X}(y \mid x) = \frac{f_{X,Y}(x,y)}{f_X(x)},\ f_X(x) > 0 \tag{2.45}$$

$f_{Y|X}(y \mid x) \geq 0,\ \forall y$ since $f_{X,Y}(x,y) \geq 0$ and $f_X(x) > 0$. In addition, if Y is a discrete random variable,

$$\sum_{\Omega_Y} f_{Y|X}(y \mid x) = \frac{\sum_{\Omega_Y} f_{X,Y}(x,y)}{f_X(x)} = \frac{f_X(x)}{f_X(x)} = 1,\ f_X(x) > 0 \tag{2.46}$$

Thus, $f_{Y|X}(y \mid x)$ is a probability mass function. In the case of a continuous random variable, the sum is replaced by an integral and $f_{Y|X}(y \mid x)$ is probability density function.

Statistical independence

Statistical independence

The random variables X and Y are said to be *statistically independent* if and only if

$$f_{X,Y}(x,y) = f_X(x)f_Y(y),\ \forall(x,y) \Leftrightarrow f_{Y|X}(y \mid x) = f_Y(y),\ \forall(x,y) \tag{2.47}$$

2.3 Characterization of a distribution

2.3.1 Moments

It is not always practical to derive the exact expression of the distribution of a random variable. Instead, distributions are often characterized by a small number of parameters, which also have a practical interpretation. These parameters are moments of different orders. We distinguish between the moments about 0, or simply *moments*, and the moments about the expected value, or *central moments*, which are invariant by translation. A probability distribution is fully described by all its moments, i.e. the $(1, \dots, \infty)$ order moments.

Let X be a continuous random variable defined on the sample space Ω.

j^{th} order moment

$$m_j = \mathrm{E}(X^j) = \int_\Omega x^j f(x)\,\mathrm{d}x,\ j \in \mathbb{N}^* \tag{2.48}$$

j^{th} order central moment

$$\mathrm{E}((X-\mu)^j) = \int_\Omega (x-\mu)^j f(x)\,\mathrm{d}x,\ j \in \mathbb{N}^* \tag{2.49}$$

where μ denotes the expected value of X

The j^{th} order moment and j^{th} order central moment of a discrete random variable are obtained by replacing the integral by a discrete sum in the above definitions.

Palm's identity

$$m_j = \mathrm{E}(X^j) = \int_0^\infty x^j f(x)\,\mathrm{d}x = \int_0^\infty j x^{j-1}(1-F(x))\,\mathrm{d}x,\ j \in \mathbb{N}^* \tag{2.50}$$

where X is a continuous non-negative real-valued random variable.

Palm's identity holds only for continuous non-negative real-valued random variables. In particular, it therefore applies to time distributions, see Section 2.4.

Example 2.7 Moments of the negative exponential distribution
If $X \sim \text{n.e.d.}(\lambda)$ then $f(x) = \lambda e^{-\lambda x},\ x \geqslant 0$ and $F(x) = 1 - e^{-\lambda x}$, see Section 2.2.3. Using (2.48), we can calculate the first, second and i^{th} moments of X.

$$\mathrm{E}(X) = \int_0^\infty x f(x)\,\mathrm{d}x = \int_0^\infty (1 - F(x))\,\mathrm{d}x = \int_0^\infty e^{-\lambda x}\,\mathrm{d}x = \frac{1}{\lambda}$$

$$\mathrm{E}(X^2) = \int_0^\infty x^2 f(x)\,\mathrm{d}x = \int_0^\infty 2x(1 - F(x))\,\mathrm{d}x = \frac{2}{\lambda}\int_0^\infty x\lambda e^{-\lambda x}\,\mathrm{d}x = \int_0^\infty 2xe^{-\lambda x}\,\mathrm{d}x$$

$$= \frac{2}{\lambda}\mathrm{E}(X) = \frac{2}{\lambda^2}$$

$$\mathrm{E}(X^i) = \int_0^\infty x^i f(x)\,\mathrm{d}x = \int_0^\infty i x^{i-1}(1 - F(x))\,\mathrm{d}x = \frac{i}{\lambda}\int_0^\infty x^{i-1}\lambda e^{-\lambda x}\,\mathrm{d}x = \frac{i}{\lambda}\mathrm{E}(X^{i-1}) = \frac{i!}{\lambda^i}$$

Note that the recursive relation between the moments:

$$\mathrm{E}(X^i) = \frac{i}{\lambda}\mathrm{E}(X^{i-1})$$

is specific to $X \sim \text{n.e.d.}(\lambda)$. It is generally not applicable to other distributions.

2.3.2 Expected value and variance

The expected value and variance are the most used moments to characterize probability distributions. There is a unique mapping from a distribution to its expected value and variance, but the reverse is not true.

Let X be a continuous random variable defined on the sample space Ω. The corresponding definitions for a discrete random variable are obtained by replacing the integral by a discrete sum.

The *expected value* or *mean*[5] of the probability distribution (or of the random variable) is the first order moment and usually denoted μ.

Expected value

$$\mu = m_1 = \mathrm{E}(X) = \int_\Omega x f(x)\,\mathrm{d}x \tag{2.51}$$

From the definition, if a and b are constant, then

$$\mathrm{E}(aX + b) = a\mathrm{E}(X) + b \tag{2.52}$$

[5]The *expected value* must not be confused with the *sample mean* presented in Chapter 8.

The *variance* is the second order central moment and is usually denoted σ^2. It tells something about the spreading of the values around the expected value, i.e. the *statistical dispersion* of the distribution.

Variance

$$\sigma^2 = \mathrm{Var}(X) = \int_\Omega (x-\mu)^2 f(x)\,\mathrm{d}x = E\left((X-\mu)^2\right) = \mathrm{E}(X^2) - \mu^2 \tag{2.53}$$

From the definition, if a is a constant, then

$$\mathrm{Var}(aX) = a^2\mathrm{Var}(X) \tag{2.54}$$

Table 2.1 sums up relevant probability distributions and provides the expected value and variance for each distribution.

Example 2.8 Expected value and variance for the Erlang distribution
Let $X \sim \mathrm{Erlang}(\lambda, k)$.

$$\mu = \mathrm{E}(X) = \int_0^\infty x f(x)\,\mathrm{d}x = \int_0^\infty x\frac{(\lambda x)^{k-1}}{(k-1)!}\lambda e^{-\lambda x}\,\mathrm{d}x = \int_0^\infty \frac{(\lambda x)^k}{(k-1)!}e^{-\lambda x}\,\mathrm{d}x$$

$$= \frac{k!}{(k-1)!}\frac{1}{\lambda}\underbrace{\int_0^\infty \frac{(\lambda x)^k}{k!}\lambda e^{-\lambda x}\,\mathrm{d}x}_{=1} = \frac{k!}{(k-1)!}\frac{1}{\lambda} = \frac{k}{\lambda}$$

$$\mathrm{Var}(X) = \int_0^\infty x^2 f(x)\,\mathrm{d}x - \mu^2 = \frac{(k+1)!}{(k-1)!}\frac{1}{\lambda^2} - \mu^2 = k(k+1)\frac{1}{\lambda^2} - \left(\frac{1}{\lambda}\right)^2 = \frac{k}{\lambda^2}$$

2.3.3 Conditional expected value

Let X and Y be two random variables defined on Ω_X and Ω_Y, respectively.

Conditional expected value

The *conditional expected value* of Y given $X = x$ is the expected value of Y computed with respect to the conditional distribution of Y given X.

$$\mathrm{E}(Y \mid X = x) = \int_{\Omega_Y} y f_{Y|X}(y \mid X = x)\,\mathrm{d}y \tag{2.55}$$

if Y is continuous, and

$$\mathrm{E}(Y \mid X = x) = \sum_{\Omega_Y} yP(Y = y \mid X = x) \tag{2.56}$$

if Y is discrete.

Table 2.1: Characteristics of relevant distributions [18]

$X \sim$	$f(x)$	$F(x)$	$\mathrm{E}(X)$	$\mathrm{Var}(X)$
n.e.d.	$\lambda e^{-\lambda x}$	$1 - e^{-\lambda x}$	$\frac{1}{\lambda}$	$\frac{1}{\lambda^2}$
Erlang-k	$\frac{(\lambda x)^{k-1}}{(k-1)!}\lambda e^{-\lambda x}$	$1 - \sum_{j=0}^{k-1} \frac{(\lambda x)^j}{j!} e^{-\lambda x}$	$\frac{k}{\lambda}$	$\frac{k}{\lambda^2}$
Gamma	$\frac{(\lambda x)^{k-1}}{\Gamma(k)}\lambda e^{-\lambda x}$	$\frac{1}{\Gamma(k)} \int_0^{\lambda x} u^{k-1} e^{-u}\,\mathrm{d}u$	$\frac{k}{\lambda}$	$\frac{k}{\lambda^2}$
Weibull	$k\lambda(\lambda x)^{k-1} e^{-(\lambda x)^k}$	$1 - e^{-(\lambda x)^k}$	$\frac{1}{\lambda}\Gamma\left(\frac{1}{k}+1\right)$	$\frac{1}{\lambda^2}\left(\Gamma\left(\frac{2}{k}+1\right) - \Gamma^2\left(\frac{1}{k}+1\right)\right)$
Uniform	$\frac{1}{b-a}$	$\frac{x-a}{b-a}$	$\frac{a+b}{2}$	$\frac{(b-a)^2}{12}$
Standard Normal, $\mathcal{N}(0,1)$	$\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$	$\Phi(x) = \int_{-\infty}^{x} \phi(u)\,\mathrm{d}u$	0	1
Normal, $\mathcal{N}(\mu,\sigma^2)$	$\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$	$\Phi\left(\frac{x-\mu}{\sigma}\right)$	μ	σ^2
Bernoulli	$\begin{cases} p & x=1 \\ 1-p & x=0 \end{cases}$		p	$p(1-p)$
Geometric	$(1-p)^x p$	$1-(1-p)^{x+1}$	$\frac{1-p}{p}$	$\frac{1-p}{p^2}$
Shifted geometric	$(1-p)^{x-1} p$	$1-(1-p)^x$	$\frac{1}{p}$	$\frac{1-p}{p^2}$
Binomial	$\binom{n}{x} p^x (1-p)^{n-x}$	$\sum_{i=0}^{x} \binom{n}{i} p^i (1-p)^{n-i}$	np	$np(1-p)$
Poisson	$\frac{\alpha^x}{x!} e^{-\alpha}$	$e^{-\alpha} \sum_{i=0}^{x} \frac{\alpha^i}{i!}$	α	α

Law of total expectation

The expected value of the conditional expected value of Y given X is equal to the expected value of Y (provided that the expected values exist).

$$\mathrm{E}_Y(Y) = \mathrm{E}_X\left(\mathrm{E}_{Y|X}(Y \mid X)\right) \tag{2.57}$$

Proof: Let X and Y be two discrete variables.

$$\begin{aligned}
\mathrm{E}_X\left(\mathrm{E}_{Y|X}(Y \mid X)\right) &= \sum_{\Omega_X} \mathrm{E}_{Y|X}(Y \mid X = x) P(X = x) \\
&= \sum_{\Omega_X} \sum_{\Omega_Y} y P(Y = y \mid X = x) P(X = x) \\
&= \sum_{\Omega_X} \sum_{\Omega_Y} y P(X = x \mid Y = y) P(Y = y) \\
&= \sum_{\Omega_Y} y P(Y = y) \underbrace{\sum_{\Omega_X} P(X = x \mid Y = y)}_{=1} \\
&= \mathrm{E}_Y(Y)
\end{aligned} \tag{2.58}$$

The theorem can be proven similarly in the case of continuous random variables, see for instance [19].

2.3.4 Measures of statistical dispersion

A first measure of statistical dispersion has been presented in Section 2.3.2, namely the variance. A more intuitive measure is the *standard deviation*, i.e. the square root of the variance. It is more understandable since it has the same unit as the observations and is hence comparable to these.

Standard deviation

$$\sigma = \sqrt{\mathrm{Var}(X)} \tag{2.59}$$

The *coefficient of variation* is a normalized, and hence dimensionless and scale free[6], measure of dispersion defined as the ratio of the standard deviation to the expected value.

[6]For a time distribution, this means that it does not depend on the time resolution (granularity).

It allows comparison of probability distributions with significantly different expected values and is often used to characterize discrete distributions.

Coefficient of variation

$$\rho = \frac{\sigma}{\mu} = \frac{\sqrt{\mathrm{E}(X^2) - \mathrm{E}(X)^2}}{\mathrm{E}(X)} \tag{2.60}$$

If $\rho > 1$, the distribution is said to be high-variance (bursty), whereas distributions with $\rho < 1$ are low-variance (non-bursty).

2.3.5 Distribution of extreme values

In some situations it is useful to know the distribution of the maximal or minimal values in n outcomes of a random variable, X. For example, there may be a requirement that the response time of a web-server must not exceed t with a given probability p.

Let $\{X_i\},\ i = 1, 2, \dots n$, be n independent and identically distributed random variables following the probability distribution $F(t) = P(X_i \leqslant t),\ i = 1, 2, \dots n$. The probability that the maximal value among the n outcomes is less than or equal to t is the probability that all the n values are less than or equal to t [16].

$$\begin{aligned} F_{\max}(t) &= P(\max\{X_i,\ i = 1, 2, \dots n\} \leqslant t) \\ &= P(\forall i,\ X_i \leqslant t) \\ &= (F(t))^n \end{aligned} \tag{2.61}$$

Similarly, the probability that the minimal value among the n outcomes is less than or equal to t is the complement of the probability that all values are greater than t.

$$\begin{aligned} F_{\min}(t) &= P(\min\{X_i,\ i = 1, 2, \dots n\} \leqslant t) \\ &= P(\exists i,\ X_i \leqslant t) \\ &= 1 - P(\forall i,\ X_i > t) \\ &= 1 - (1 - F(t))^n \end{aligned} \tag{2.62}$$

Note that if the variables are not identically distributed, the individual distributions must be used.

2.3.6 Measures of dependence between variables

The notion of dependence between two random variables is defined in Section 2.1.4. This section focuses on how to quantify the dependence.

Covariance

The *covariance* between two random variables X and Y is defined as:

Covariance

$$\sigma_{XY} = \mathrm{Cov}(X,Y) = \mathrm{E}((X-\mu_X)(Y-\mu_Y)) = \mathrm{E}(XY) - \mu_X\mu_Y \tag{2.63}$$

From the definition, it follows that $\mathrm{Cov}(X,X) = \mathrm{Var}(X)$ and that the covariance is a symmetric function, i.e. $\mathrm{Cov}(X,Y) = \mathrm{Cov}(Y,X)$.

The covariance gives an indication of how two variables vary together but does not convey an easy interpretation of the strength of the dependence since it is not scale free and depends on the unit of both X and Y. The unit of the covariance is the product of the units of the random variables.

The covariance tends to be positive if the two variables often differ from their mean in the same direction, i.e. either both positively or both negatively, and negative if they differ from their mean in opposite directions. The more often they differ from their mean in the same direction (resp. in opposite directions), the more positive (resp. negative) the covariance. If the covariance is zero, the variables are *uncorrelated*. The covariance of independent variables is zero, but a zero covariance does not necessarily imply that the variables are independent.

Example 2.9 Dependence with zero covariance
It is discovered that two noise signals, denoted X and Y, have a dramatic impact on the signal quality on a transmission line. A large number of values of X and Y is collected to determine whether they come from the same source of noise or not. Naturally, the dependence between the two signals is studied. Calculations show that $\sigma_{XY} = 0$. So the two signals are deemed to be independent and to come from two separate sources. One noise source is found and removed. But then both signals have vanished! Has the other source vanished meanwhile or was there only a single noise source? After further investigations, it is finally found that the signal Y is a quadratic amplification of signal X, i.e. $Y = X^2$. This means that the two signals are (functionally) dependent.
To see how this is possible, assume that the noise signal X follows a standard normal distribution, $X \sim \mathcal{N}(0,1)$, which is a very common way to model white noise. The probability density function is therefore:

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$$

The covariance is calculated using (2.63).

$$\sigma_{XY} = \mathrm{E}(XY) - \mu_X\mu_Y$$

and, since $X \sim \mathcal{N}(0,1)$, $\mu_x = \mathrm{E}(X) = 0$ and $\mu_y = \mathrm{E}(X^2) = 1$. Hence,

$$\sigma_{XY} = \mathrm{E}(XX^2) = \mathrm{E}(X^3) = \int_{-\infty}^{\infty} x^3 \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}\,\mathrm{d}x = 0$$

The last equality comes directly by noticing that $x^3 \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$ is an odd function.
Therefore the covariance of X and Y does equal 0. Remember that X and Y may not be independent although their covariance is zero.

Correlation coefficient

As mentioned, the covariance does not indicate anything regarding the strength of the dependence between two random variables. The *correlation coefficient*, or simply *correlation*, is a dimensionless (scale-free) measure and thus indicates both the direction and strength of the dependence [2].

Correlation coefficient

$$\rho_{XY} = \frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{Var}(X)\cdot\mathrm{Var}(Y)}} = \frac{\sigma_{XY}}{\sigma_X\sigma_Y} \tag{2.64}$$

The correlation coefficient satisfies the inequality $-1 \leqslant \rho_{XY} \leqslant 1$. If $\rho_{XY} = 0$, X and Y are uncorrelated (but not necessarily independent). If $\rho_{XY} > 0$ or $\rho_{XY} < 0$, X and Y are positively or negatively correlated, respectively. This means that they differ from their mean in the same direction, or in opposite directions, respectively. The closer the coefficient is to either -1 or 1, the stronger the correlation between the two variables.

Example 2.10 Error in the address-name mapping list in a DNS server
There is an error in the address-name mapping list in a DNS server. All the clients that update their list based on this list become "infected".
Let X= "the client A has an erroneous mapping" and Y= "the central DNS server is faulty".
The correlation coefficient is positive, $\rho_{XY} > 0$, which means that if the client A is erroneous, then the central server most likely also is.

2.3.7 Sum of random variables

Let S be a sum of random variables X_i, $S = \sum_{i=1}^n X_i$.

Expected value of a sum of random variables

The expected value of the sum S of random variables X_i, $S = \sum_{i=1}^n X_i$, is equal to the sum of the expected values.

$$\mu_S = \mathrm{E}(S) = \sum_{i=1}^{n} \mathrm{E}(X_i) \tag{2.65}$$

From the definition of the variance (2.53),

$$\begin{aligned}
\mathrm{Var}(S) &= \mathrm{E}\left((S-\mu_s)^2\right) \\
&= \mathrm{E}\left(\left(\sum_{i=1}^{n} X_i - \sum_{i=1}^{n}\mathrm{E}(X_i)\right)^2\right) = \mathrm{E}\left(\left(\sum_{i=1}^{n}(X_i - \mathrm{E}(X_i))\right)^2\right) \\
&= \mathrm{E}\left(\sum_{i=1}^{n}(X_i - \mathrm{E}(X_i)) \cdot \sum_{j=1}^{n}(X_j - \mathrm{E}(X_j))\right) \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n}\mathrm{E}\left((X_i - \mathrm{E}(X_i))(X_j - \mathrm{E}(X_j))\right) \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n}\mathrm{Cov}(X_i, X_j)
\end{aligned} \tag{2.66}$$

Variance of a sum of random variables

The variance of the sum S of random variables X_i, $S = \sum_{i=1}^{n} X_i$, is:

$$\sigma_S^2 = \mathrm{Var}(S) = \sum_{i=1}^{n}\sum_{j=1}^{n}\mathrm{Cov}(X_i, X_j) \tag{2.67}$$

If the X_i's are uncorrelated, $\forall i, j,\ i \neq j, \mathrm{Cov}(X_i, X_j) = 0$, and therefore

Variance of a sum of uncorrelated random variables

The variance of the sum S of uncorrelated random variables X_i, $S = \sum_{i=1}^{n} X_i$, is:

$$\sigma_S^2 = \mathrm{Var}(S) = \sum_{i=1}^{n}\mathrm{Var}(X_i) \tag{2.68}$$

Example 2.11 Expected value and variance for the Erlang distribution
The calculation of the expected value and variance simplifies using (2.65) and (2.68) compared to Example 2.8. The Erlang-k distribution is the distribution of the sum of k independent random variables $X_i \sim \text{n.e.d.}(\lambda)$. Using the expected value and variance of the negative exponential distribution, see Table 2.1:

$$\mathrm{E}(S) = \sum_{i=1}^{k}\mathrm{E}(X_i) = k\mathrm{E}(X) = \frac{k}{\lambda}$$

$$\mathrm{Var}(X) = \sum_{i=1}^{k}\mathrm{Var}(X_i) = k\mathrm{Var}(X) = \frac{k}{\lambda^2}$$

2.4 Time distributions

This section addresses *time distributions*, i.e. distributions of continuous non-negative ($t \geqslant 0$) real-valued random variables representing times. Examples include:

- times to and between events, e.g. the time to the next failure or the time between consecutive failures,
- durations of activities, e.g. the repair time or the call duration, and
- sojourn times, e.g the time in a queue.

In particular, this section focuses on residual lifetime distributions, i.e. distributions of the time to the next event. We distinguish between the residual lifetime given a known operation time and from a random point of time. For instance, the residual lifetime, literally speaking, of a second-hand mobile phone is different depending on whether its age is known or not. For further details see [16, 17].

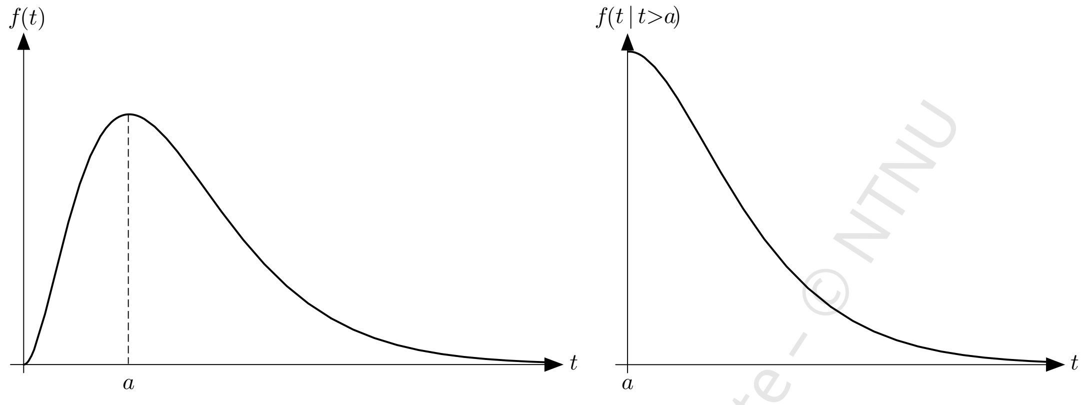
Example 2.12 $2 + 2 = 2$, a paradox arises
We would like to study how failures occur in a web-server which is known to be prone to failure. From monitoring it over a long period, it is known that the time between failures is on average two hours. We start observing the server at a random point of time. First, we check the time the last failure occurred and find out that it was almost two hours ago. This looks good, we should not wait too long before the next failure occurs. After waiting two hours for the next failure, we think we have had really bad luck. Hence, we decide to have a closer look and repeat the experiment several times with the same web-server; that is at a random time we check the time the last failure occurred and wait until the next failure occurs. Surprisingly, the results show that we were actually not "unlucky" the first time we performed the experiment. The mean time since the last failure is two hours, so is the mean time we have to wait before the next failure occurs and so is the mean time between failures! We must have done a mistake!
This apparent paradox is due to a specific distribution of the time to the next failure and is resolved in Example 2.14.

2.4.1 Residual lifetime

The *residual lifetime* is the lifetime under the condition that a certain age a has been reached.

Let T represent the lifetime. The cumulative distribution function of the residual lifetime is the conditional distribution $F(t + a \mid a)$. Given that $t \geqslant 0$ and assuming that $P(T > a) > 0$, we have:

$$\begin{aligned} P(T > t + a \mid T > a) &= \frac{P(T > t + a \cap T > a)}{P(T > a)} \\ &= \frac{P(T > t + a)}{P(T > a)} = \frac{1 - F(t + a)}{1 - F(a)} \end{aligned} \tag{2.69}$$



(a) Unconditional density function **(b) Residual lifetime density function**

Figure 2.5: Probability density functions of the lifetime

and thus:

$$F(t+a \mid a) = 1 - P(T > t+a \mid T > a) = 1 - \frac{1-F(t+a)}{1-F(a)} = \frac{F(t+a)-F(a)}{1-F(a)} \tag{2.70}$$

The probability density function of the residual lifetime is:

$$f(t+a \mid a) = \frac{\mathrm{d}}{\mathrm{d}t} F(t+a \mid a) = \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{F(t+a)-F(a)}{1-F(a)}\right) = \frac{f(t+a)}{1-F(a)} \tag{2.71}$$

Note that the conditional probability density function $f(t+a \mid a)$ can be viewed as a rescaling of the unconditional probability density function $f(t)$ in the domain of values greater than or equal to a. The scaling factor is the probability that the lifetime is greater than a. In Figure 2.5 both the unconditional, $f(t)$, and the conditional, $f(t \mid a)$, probability density functions are shown.

The expected residual lifetime is obtained by replacing $f(x)$ by $f(t+a \mid a)$ in (2.51) and using Palm's identity (2.50).

$$\begin{aligned}\mu_{T|a} = \mathrm{E}(T \mid a) &= \int_0^\infty t f(t+a \mid a)\,\mathrm{d}t = \frac{1}{1-F(a)}\int_0^\infty t f(t+a)\,\mathrm{d}t \\ &= \frac{1}{1-F(a)}\int_0^\infty (1-F(t+a))\,\mathrm{d}t\end{aligned} \tag{2.72}$$

The probability that the lifetime terminates in an infinitely small interval $\mathrm{d}t$, $\mathrm{d}t \to 0$, given that the time t has been reached is called the *death rate* at time t and is denoted $\phi(t)$. It is obtained by letting $a = t$ and $t = \mathrm{d}t$ in (2.71).

$$\phi(t) = \lim_{\mathrm{d}t\to 0} \frac{P(t < T \leqslant t+\mathrm{d}t \mid T > t)}{\mathrm{d}t} = \lim_{\mathrm{d}t\to 0} f(t+\mathrm{d}t \mid t) \tag{2.73}$$

Death rate

$$\phi(t) = \frac{f(t)}{1 - F(t)} \tag{2.74}$$

In the dependability theory context, it is known as the *failure rate*, see Chapter 7.

Example 2.13 Waiting time distributions
A printer is connected to a local network. The administrator is interested to know how long a job is queued before the printing starts. Jobs sent to the printer while the queue is empty do not wait.

The time distribution of all the jobs that have to wait is the residual waiting time distribution given that $t > 0$.

Let W denote the waiting time for a job.

The distribution of the waiting time for all jobs is $F_W(t) = P(W \leqslant t)$, while the distribution for the jobs given that they have to wait is:

$$F_{W|W>0}(t) = P(W \leqslant t \mid W > 0) = \frac{F_W(t) - F_W(0)}{1 - F_W(0)}$$

The portion of jobs that have to wait is: $P_W = 1 - F_W(0) = P(W > 0)$.
So we can write: $P_W(1 - F_{W|W>0}(t)) = 1 - F_W(t)$, which gives:

$$P_W f_{W|W>0}(t)\,\mathrm{d}t + (1 - P_W)\,\delta(t)\,\mathrm{d}t = f_W(t)\,\mathrm{d}t$$

where $\delta(t)$ denotes the Dirac delta function.

The expected waiting time in the printing queue is therefore:

$$\mathrm{E}(W) = \int_0^\infty t f_W(t)\,\mathrm{d}t = \int_0^\infty t P_W f_{W|W>0}(t)\,\mathrm{d}t + \underbrace{\int_0^\infty t(1 - P_W)\,\delta(t)\,\mathrm{d}t}_{=0}$$

$$= P_W \int_0^\infty t f_{W|W>0}(t)\,\mathrm{d}t = \mathrm{E}(W \mid W > 0) P_W$$

The same result can be obtained using Palm's identity (2.50):

$$\mathrm{E}(W) = \int_0^\infty (1 - F_W(t))\,\mathrm{d}t = P_W \int_0^\infty (1 - F_{W|W>0}(t))\,\mathrm{d}t = \mathrm{E}(W \mid W > 0) P_W$$

or using the law of total expectation (2.57):

$$\mathrm{E}(W) = \mathrm{E}(W \mid W > 0) P_W + \underbrace{\mathrm{E}(W \mid W = 0)(1 - P_W)}_{=0} = \mathrm{E}(W \mid W > 0) P_W$$

or directly using (2.72):

$$\mathrm{E}(W \mid W > 0) = \frac{1}{P_W} \int_0^\infty (1 - F_W(t))\,\mathrm{d}t \Leftrightarrow \mathrm{E}(W) = \mathrm{E}(W \mid W > 0) P_W$$

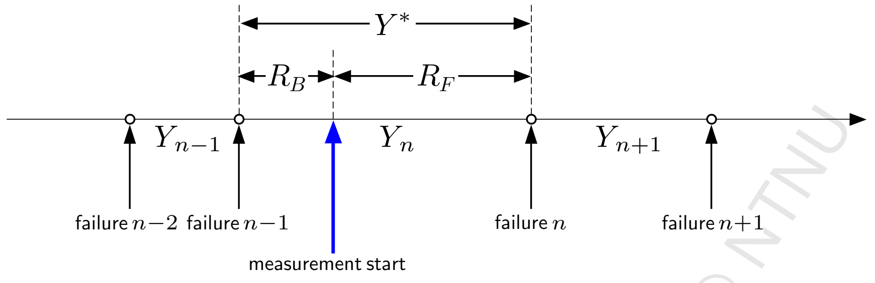


Figure 2.6: Time to the next failure from a random point of time

2.4.2 Forward recurrence time

In the previous section, we studied the distribution of the residual lifetime given that the time is greater than a. We will now focus on the residual lifetime from a random point of time and see how the expressions derived previously are modified. This time is referred to as the *forward recurrence time* and corresponds to the time we measure in Example 2.12; at a random time, we start measuring the time to the next failure. We will use this example in the rest of this section.

Let Y be a random variable representing the time between two consecutive failures, Y^* the length of the random time interval in which we observe the time to the next failure, i.e. the forward recurrence time, R_F, see Figure 2.6.

1. The time intervals Y_i between failures are identically distributed variables.

$$f_Y(y)\,\mathrm{d}y = P(y < Y \leqslant y + \mathrm{d}y),\; y \geqslant 0 \tag{2.75}$$

2. The probability that we start measuring the time to the next failure in the interval Y_i is proportional to the length of the interval (length-biased sampling). The probability that the measurement start is in the interval Y^* of length y is therefore:

$$\begin{aligned} & f_{Y^*}(y)\,\mathrm{d}y = P(y < Y^* \leqslant y + \mathrm{d}y) \propto y f_Y(y)\,\mathrm{d}y \\ \Leftrightarrow \quad & f_{Y^*}(y)\,\mathrm{d}y = C y f_Y(y)\,\mathrm{d}y \end{aligned} \tag{2.76}$$

where C denotes a constant.

$$f_{Y^*}(y)\,\mathrm{d}y = C y f_Y(y)\,\mathrm{d}y \Leftrightarrow \int_0^\infty f_{Y^*}(y)\,\mathrm{d}y = C\int_0^\infty y f_Y(y)\,\mathrm{d}y \Leftrightarrow C = \frac{1}{\mathrm{E}(Y)} \tag{2.77}$$

Thus

$$f_{Y^*}(y)\,\mathrm{d}y = \frac{1}{\mathrm{E}(Y)} y f_Y(y)\,\mathrm{d}y \tag{2.78}$$

3. The measurement points are chosen randomly in time so the forward recurrence time is uniformly distributed over the interval. The conditional probability density function of R_F given that a measurement is done in the interval Y^* of length y is thus given by:

$$f_{R_F|Y^*}(t \mid Y^* = y)\,\mathrm{d}t = P(t < R_F \leqslant t + \mathrm{d}t \mid Y^* = y) = \frac{1}{y}\,\mathrm{d}t,\ 0 < t \leqslant x \qquad (2.79)$$

4. Using the law of total probability (2.9), we find the probability density function of the forward recurrence time:

$$f_{R_F}(t) = \int_t^{\infty} f_{R_F|Y^*}(t \mid Y^* = y) f_{Y^*}(y)\mathrm{d}y = \int_t^{\infty} \frac{1}{y}\frac{1}{\mathrm{E}(Y)} y f_Y(y)\,\mathrm{d}y = \frac{1 - F_Y(t)}{\mathrm{E}(Y)} \qquad (2.80)$$

The expected forward recurrence time is calculated using Palm's identity (2.50):

$$\begin{aligned}\mathrm{E}(R_F) &= \int_0^{\infty} t f_{R_F}(t)\,\mathrm{d}t = \int_0^{\infty} t\frac{1 - F_Y(t)}{\mathrm{E}(Y)}\,\mathrm{d}t \\ &= \frac{1}{2\mathrm{E}(Y)}\int_0^{\infty} 2t(1 - F_Y(t))\,\mathrm{d}t = \frac{\mathrm{E}(Y^2)}{2\mathrm{E}(Y)}\end{aligned} \qquad (2.81)$$

Example 2.14 $2 + 2 = 2$, the paradox resolved (Example 2.12 continued.)
Studies have shown that time between failures in the web-server presented in Example 2.12 is negatively exponentially distributed, $T_{BF} = Y \sim \text{n.e.d.}(\lambda)$. The two first moments of T_{BF} are, see Table 2.1: $MTBF = \mathrm{E}(T_{BF}) = \frac{1}{\lambda}$ and $\mathrm{E}(T_{BF}^2) = \frac{2}{\lambda^2}$.
Using (2.81), the expected time to the next failure, i.e. the expected forward recurrence time, is:

$$\mathrm{E}(R_F) = \frac{\mathrm{E}(T_{BF}^2)}{2\mathrm{E}(T_{BF})} = \frac{\frac{2}{\lambda^2}}{2 \cdot \frac{1}{\lambda}} = \frac{1}{\lambda}$$

that is, when $T_{BF} \sim \text{n.e.d.}(\lambda)$, $\mathrm{E}(R_F) = MTBF$.
Moreover, $F_{T_{BF}}(t) = 1 - e^{-\lambda t}$ and, using (2.80), we have:

$$f_{R_F}(t) = \frac{1 - F_{T_{BF}}(t)}{\mathrm{E}(T_{BF})} = \lambda e^{-\lambda t}$$

that is, when $T_{BF} \sim \text{n.e.d.}(\lambda)$, $R_F \sim \text{n.e.d.}(\lambda)$.
The paradox in Example 2.12 was thus not a paradox, but an illustration of the memoryless property of the negative exponential distribution (see Section 5.1.3). The probability of terminating is independent of the actual age or, in other words, the failure rate is constant.
Note that the time to the next failure, or forward recurrence time, R_F is different from the *time to failure* T_F defined on page 28 which is the time to the next failure given that the system is working. Only if the down time is infinitely short, $T_D = 0$, then the time between failures is equal to the up time, $T_{BF} = T_U$, and $T_F = R_F$.

Along a similar line of arguments, it can be shown that the distribution of the backward recurrence time R_B, i.e. the time from a random point in time and backwards to the last event, see Figure 2.6, is equal to the distribution of the forward recurrence time R_F, $f_{R_B} = f_{R_F}$

3 Fundamentals of stochastic processes

In Chapter 2 we have seen how to characterize random variables, but we have only considered one random variable at a time, e.g. the packet inter-arrival time to a router. For a whole system, we have to consider series of events that influence several random variables, i.e. *stochastic processes*. For instance, a series of packet arrivals is a stochastic process. A more complex process is the series of arrivals on all the incoming links to a router, how many and how long packets stay in the different buffers, etc.

This chapter gives a general presentation of stochastic processes. In particular, Section 3.4 gives an important and generally applicable result, namely Little's formula. A specific class of memoryless processes, namely *Markov*[1] *processes*, is detailed in Chapter 5. The future of a Markov process only depends on the current state of the process and not on what has occurred before. As an example, if the future behavior of a router at any instant depends on the number of packets in the buffers but not on how the buffers have been filled, it may be modeled as a Markov process. Markov processes are very useful for mathematical dependability and performance analysis, but typically require some idealized assumptions.

3.1 Types of processes

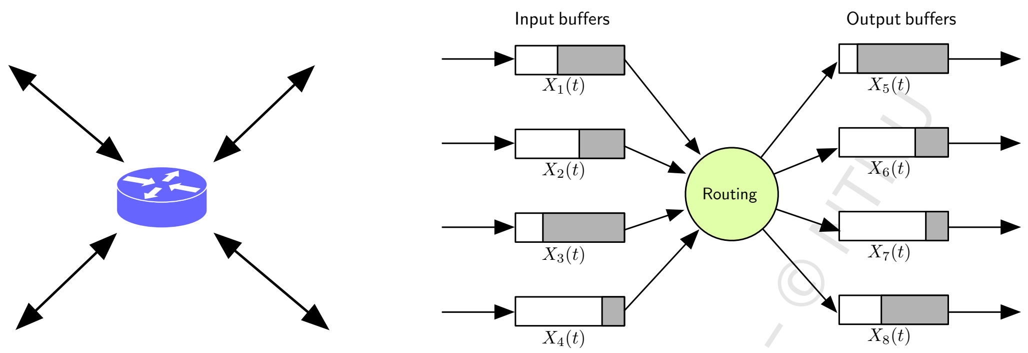
A *stochastic process* is defined as a set of random variables $\underline{X}(t)$ (vector function of t). In the context of ICT-systems, t always represents the time. The set of all possible values of $\underline{X}(t)$ is called the *state-space* and is denoted Ω, i.e. $\underline{X}(t) \in \Omega$. The value of $\underline{X}(t)$ describes the state of the system.

The number of random variables is called the *dimension* of the process, i.e. if n denotes the dimension of the process then $\underline{X}(t) = \{X_1(t), X_2(t), \dots, X_n(t)\}$. Most of the processes discussed in this book are one-dimensional, and we then write $X(t) \in \Omega$.

When a stochastic process is a series of events, it is also common to say that the events are generated by the process.

Example 3.1 Router
Consider the router shown in Figure 3.1(a). Assume that the router has input and output buffers, where the input buffers contains the packets received but not yet routed, and the output buffers the packets routed but not yet forwarded. For the sake of simplicity, assume that a packet is transferred

[1]Andrey Andreyevich Markov (1856-1922). Russian mathematician.



(a) Router with four links **(b) Simple performance model**

Figure 3.1: The stochastic process $\underline{X}(t)$ describes the buffer filling level in the router

instantaneously from an input buffer to an output buffer. The state of the system is defined by the number of packets in each buffer, $\underline{X}(t) = \{X_1(t), X_2(t), \ldots, X_8(t)\}$. Assume that each buffer corresponds to a separate physical storage of length L, e.g. $L = 0.5$ Mbytes. The state-space is then $\Omega = L^8$.

3.1.1 Processes in time and space

A stochastic process has a time dimension, which describes the progress of the process, and one or more space dimensions, which describe the states of the process. The states of the process may be defined by the utilization of resources, the number of pending service requests, which parts of the system are working or failed etc.

We distinguish between processes continuous or discrete in time, and continuous or discrete in space, depending on the physical system but also on the study to be carried out.

Discrete and continuous time

In the real-world, the time is continuous. In most of the models and analysis the time is therefore considered to be continuous, as illustrated for example in Figure 3.2. However, in some cases the study is simplified if discrete time, i.e. a series of points in time, $t = 0, 1, 2, \ldots, \infty$, is used.

- *Systems with discrete time.* For some systems the time is partitioned, e.g. for a slotted ring or for the radio interface between a mobile station and a base station in a GSM system. In these systems, events of interest for a performance modeling can be associated with a particular timeslot or frame. The time can therefore be

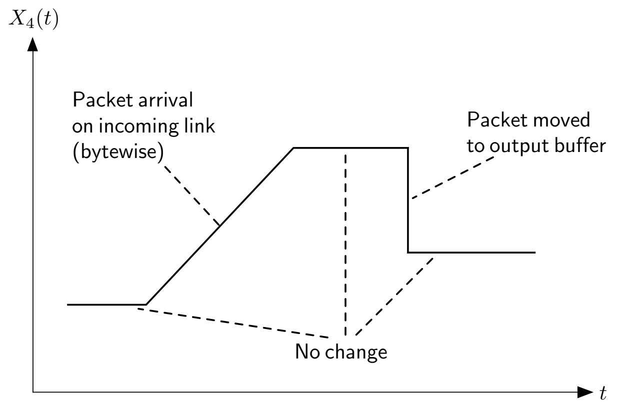

$X_4(t)$
Packet arrival on incoming link (bytewise)
Packet moved to output buffer
No change
t

Figure 3.2: Example of evolution of the filling level of buffer 4 in Figure 3.1

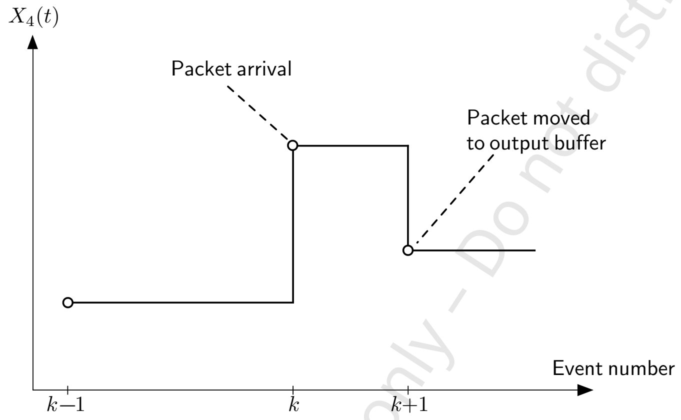
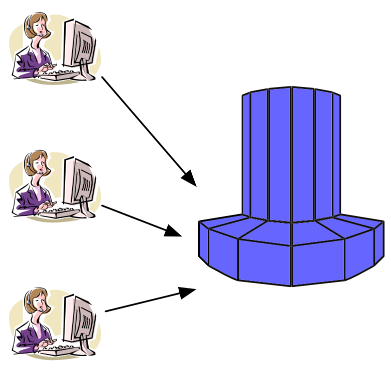


Figure 3.3: Example of discrete time process embedded in the continuous time shown in Figure 3.2

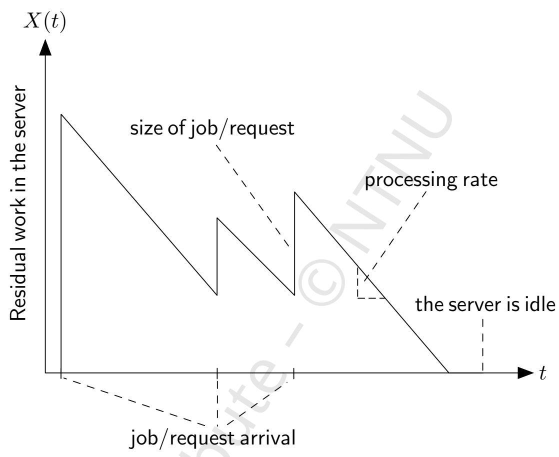
the number of timeslots or frames elapsed. Note that, in most of the systems, the timeslots or frames have a constant length so the discrete time is proportional to the real time.

- *Embedded processes.* For the mathematical analysis of systems, it may be relevant to consider the state of the system only when a particular type of events occurs. The time is then represented by the number of events of this type that have occurred. For example in Figure 3.2, the discrete time can be defined as the points of time when the content of the buffer changes. The corresponding discrete time process is shown in Figure 3.3. Note that any type of events that will advance the discrete time can be chosen and that the time between these events may be a random variable.

  Analysis based on embedded processes is beyond the scope of this book.



(a) Server system



(b) Illustration of the stochastic process that gives the residual work in the server

Figure 3.4: Residual work

Discrete and continuous state space

Similarly to the time space, the state space may be discrete or continuous. In reality, the state of an ICT-system is always discrete[2]. However, it may be sometimes easier to model discrete states with high granularity using a continuous variable. Examples of such approximate continuous space variables include:

- *Buffer filling level.* This is illustrated in Figures 3.1, 3.2 and 3.3. The granularity is of one byte. With respect to typical values of packet sizes and buffer lengths, it is thus natural to consider the buffer filling level as a continuous variable.
- *Residual work.* This example is illustrated in Figure 3.4. A number of users send service requests/jobs to a server. The requests/jobs in the system that have not yet been completely served constitute the residual work. Work in this context refers to the time it takes to serve the request/perform the job. As long as there are requests/jobs waiting for service, the residual work decrease in time according to the processing rate.

  Note that the residual work seen as state variable depends neither on the priorities of the different requests, nor on the sequencing, etc.

In discrete state space systems, the state of the system is given by the state of the system components. For instance, the state of the router in Figure 3.1 is determined by the combined states of each single buffer. A system component with influence on the system state is a resource, see Section 1.1.1. Examples of resources and states are timeslots in

[2]Wave propagation and analog sources like speech are not considered.

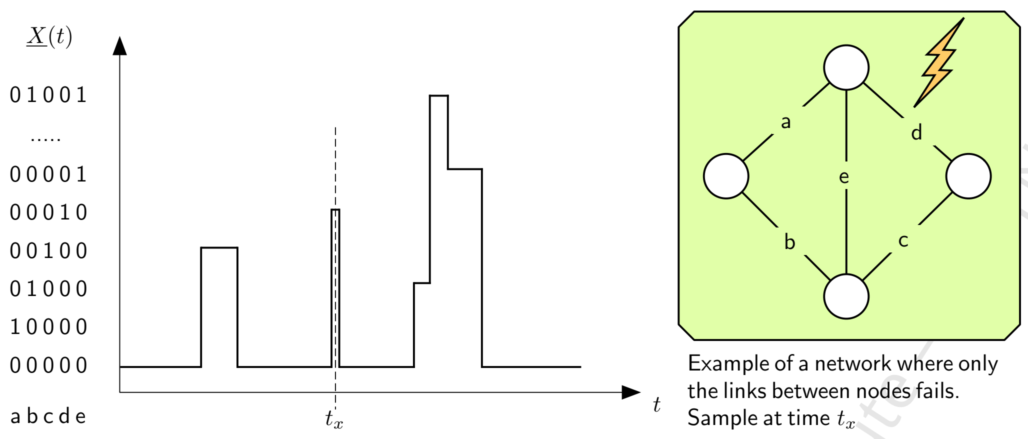

$\underline{X}(t)$
01001
.....
00001
00010
00100
01000
10000
00000
a b c d e
t_x
t
a
d
e
b
c
Example of a network where only the links between nodes fails. Sample at time t_x

Figure 3.5: Failure states in a network

a time-division system that can be idle or busy, a transmission channel that can be idle or busy, a server node in a network that can be intact or in one particular failed state. Examples of discrete state spaces include:

- *Failure states in a network.* A very simple 5 link network where only the links may fail and where the links only have two possible states, working (0) or failed (1), is shown in Figure 3.5. The state of the system is a stochastic process $\underline{X}(t)$, and the state-space Ω is constituted by the 2^5 possible combination of link failures.

- *Number of resources in use,* in the case of a group of identical resources where only the number of busy resources matters, e.g. the number of busy slots in a synchronous time-division system, or the number of speech generators in use in a group. See for example Figure 1.14.

3.1.2 Point processes

A *point process* is a process that only describes the occurrence of events, regardless their type. A point process is a one-dimensional process which state is defined by the number of events that have occurred, i.e. $X(t) = N(t)$ where $N(t)$ denotes the number of events in the interval $(0, t]$. In the following, the inter-event times, denoted Y_k, are referred to as the *increments*. This is to distinguish them from the absolute time t. Examples of point processes include software failures in a PC and calls to an emergency number.

A point process is *regular* if and only if at most one event occurs at a time, i.e. $Y_k \neq 0,\ \forall k$. See Figure 3.6 for an illustration.

Regular point process

A point process is *regular* if and only if at most one event occurs at a time.

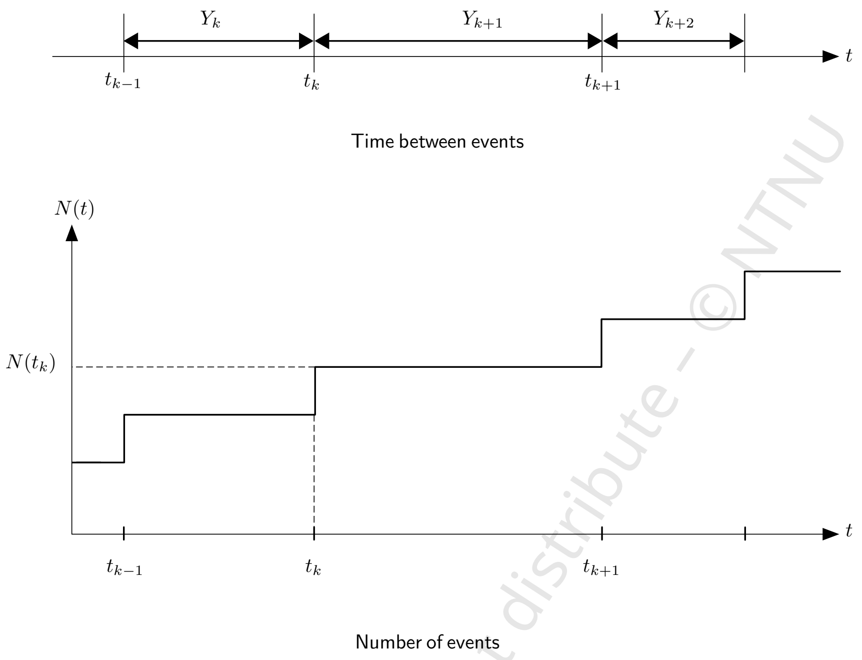


Figure 3.6: Illustration of a regular point process

Relation between the number of events and the time to an event

Let S_k be the time to the k^{th} event, $k \in \mathbb{N}^*$. For a regular point process, by definition

$$S_k = \sum_{i=1}^{k} Y_i,\ k \in \mathbb{N}^* \tag{3.1}$$

Then, $N(t) < k$ if and only if $S_k > t$.

Relation between the number of events and the time to an event

For a regular point process, the probability that the number of events in the interval $(0, t]$ is less than $k \in \mathbb{N}^*$ is equal to the probability that the k^{th} event occurs after time t.

$$P(N(t) < k) = P(S_k > t),\ k \in \mathbb{N}^* \tag{3.2}$$

Using (3.2), the probability that an event occurs after a time interval Δt can be determined if the probability of the number of events in $(t, t + \Delta t]$ is known.

Example 3.2 Quality of Service
A football match is broadcasted by satellite. The service level agreement for the broadcast allows at most one disruption during the football match. If several disruptions occur, the satellite operator has to reduce the price by 10%. What is the probability that the price is not reduced?

The satellite operator knows that the number of disruptions over a period of time follows a Poisson distribution, see Table 2.1 on page 71, with parameter $\lambda\tau$ where $\lambda = 0.4$ [disruption per hour] is the failure intensity and $\tau = 2$ hours the broadcast duration, i.e. the expected number of disruptions during the match is 0.8.
Using (3.2), the probability that the second disruption occurs after the broadcast is the probability that 0 or 1 disruption only occurs during the match.

$$\begin{aligned} P(\text{no price reduction}) &= P(N(\tau) < 2) = P(S_2 > \tau) \\ &= e^{-\lambda\tau} + \frac{\lambda\tau}{1!}e^{-\lambda\tau} = \sum_{i=0}^{1} \frac{(\lambda\tau)^i}{i!}e^{-\lambda\tau} \\ &= 1 - F_{S_2}(\tau) \end{aligned}$$

where $F_{S_2}(\tau) = 1 - \sum_{i=0}^{1} \frac{(\lambda\tau)^i}{i!} e^{-\lambda\tau}$. Hence, the time to the second disruption is Erlang-2 distributed with parameter λ.
Numerically we have: $P(\text{no price reduction}) = 0.45 + 0.36 = 0.81$.
More generally, this example illustrates the relation between the Erlang-k probability density function and the Poisson cumulative distribution function given in Table 2.1 on page 71. More on this in Chapter 5.

$$\int_0^t \frac{(\lambda x)^{k-1}}{(k-1)!}\lambda e^{-\lambda x}\,\mathrm{d}x = 1 - \sum_{j=0}^{k-1} \frac{(\lambda t)^j}{j!}e^{-\lambda t}$$

Intensity

Intensity

Expected number of events in a time interval divided by the interval length Δt when $\Delta t \to 0$.

$$z(t) = \lim_{\Delta t \to 0} \frac{\mathrm{E}(N(t+\Delta t) - N(t))}{\Delta t} \tag{3.3}$$

where $N(t_2) - N(t_1)$ is the number of events in $(t_1, t_2]$ given that $t_1 < t_2$. Note that in (3.3) neither the events that have occurred between 0 and t nor the number of events $N(t)$ are known.

The intensity has the dimension [time^{-1}]. It is often a parameter for traffic and dependability processes and sometimes takes an application-specific name, for instance:

- BHCA (Busy Hour Call Attempts): Call intensity during the busy hour, see page 34; used for telephone systems.
- Fit, Fits (Failure units): Failures/(10^9 hours); used to indicate the failure intensity (and the failure rate if it is constant) of hardware components.

In the case of a regular point process, at most one event occurs at a time. That is, when $\Delta t \to 0$, $N(t+\Delta t) - N(t)$ is equal to 0 or 1, or, in other words, the number of events in the interval $(t, t+\Delta t]$ is Bernoulli distributed. Thus,

$$\mathrm{E}(N(t+\Delta t) - N(t)) = P(N(t+\Delta t) - N(t) = 1) \tag{3.4}$$

Intensity of a regular point process

Probability that an event occurs during a short time interval divided by the interval length Δt when $\Delta t \to 0$.

$$z(t) = \lim_{\Delta t \to 0} \frac{P(N(t+\Delta t) - N(t) = 1)}{\Delta t} \tag{3.5}$$

Since there is at most one event at a time, we can write

$$\begin{aligned} P(N(t+\Delta t) - N(t) \geqslant 2) &= o(\Delta t) \\ P(N(t+\Delta t) - N(t) = 1) &= z(t)\Delta t + o(\Delta t) \\ P(N(t+\Delta t) - N(t) = 0) &= 1 - z(t)\Delta t + o(\Delta t) \end{aligned} \tag{3.6}$$

where $\pm o(\Delta t)$ is the probability that two or more events occur in the interval $(t, t+\Delta t]$ and tends to 0 faster than Δt, i.e. $\lim_{\Delta t \to 0} \frac{o(\Delta t)}{\Delta t} = 0$.

Rate

The intensity is often confused with the *rate* $\phi(y)$. The rate is related to the time to the next event given that the time at which the last event occurred is known, i.e. to the increment Y_k. For the sake of simplicity, the index k is omitted in the rest of this section.

Rate

Probability that an event occurs during a short time interval $(y, y+\Delta y]$ given that the event has not occurred before y divided by the interval length Δy when $\Delta y \to 0$.

$$\phi(y) = \lim_{\Delta y \to 0} \frac{P(y < Y \leqslant y + \Delta y \mid Y > y)}{\Delta y} \tag{3.7}$$

Since $(y < Y \leqslant y + \Delta y) \cap (Y > y) \Leftrightarrow (y < Y \leqslant y + \Delta y)$, using (2.6) and the definition of a cumulative and probability distribution functions given in Section 2.2.2 and, we can write

$$\begin{aligned} \phi(y) &= \lim_{\Delta y \to 0} \frac{1}{\Delta y} \frac{P((y < Y \leqslant y + \Delta y) \cap (Y > y))}{P(Y > y)} \\ &= \lim_{\Delta y \to 0} \frac{1}{\Delta y} \frac{P(y < Y \leqslant y + \Delta y)}{P(Y > y)} \\ &= \frac{f_Y(y)}{1 - F_Y(y)} \end{aligned} \tag{3.8}$$

Example 3.3 Dependability concepts
If we observe a system, for instance a workstation, composed of a large number of components which may fail due to different causes and where failures are rectified, then the failures form a point process and we are interested in the failure intensity $z(t)$. The failure intensity is relevant because it says something about the probability that the system fails immediately, regardless the failures that occurred previously.
On the other hand, if we consider a single component, for instance the screen, which is an essential component of a workstation, and study its life-cycle, then we are interested in the lifetime distribution, and thus we consider the failure rate $\phi(t)$ for the component.

Note that λ is commonly used indifferently to denote both the failure rate and the failure intensity which contributes to the confusion.

Renewal processes

Renewal process

A point process is a *renewal process* if and only if the inter-event times are independent and identically distributed.

The intensity of a renewal process tends to the inverse of the expected inter-event time (increment).

$$\lim_{t\to\infty} z(t) = \frac{1}{\mathrm{E}(Y)} \tag{3.9}$$

The expected number of events is called the *renewal function.*

$$H(t) = \mathrm{E}(N(t)) \tag{3.10}$$

For large values of t, it can be shown [20] that

$$H(t) = \frac{t}{\mathrm{E}(Y)} + \frac{\mathrm{Var}(Y) - \mathrm{E}(Y)^2}{2\mathrm{E}(Y)^2} + o(1) \tag{3.11}$$

The list below gives the letters commonly used for renewal processes. The last symbol GI only says that it is a renewal process.

M	Markov, negatively exponentially distributed increments (i.e. events occur according to a Poisson process, see Section 5.1)
D	deterministic increments
E_k	Erlang-k distributed increments
H_k	hyper-exponentially distributed increments
GI	general and independent increments

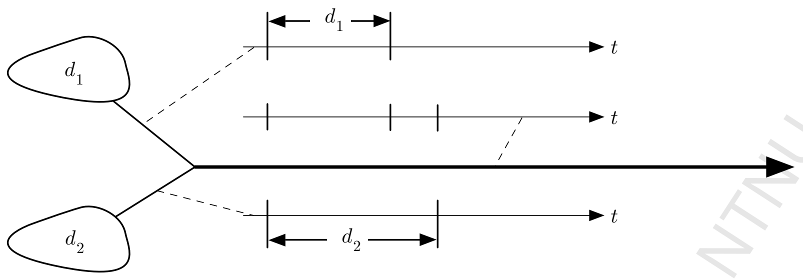


Figure 3.7: Merging of two deterministic processes

Merging renewal processes will generally not result in a new renewal process. In Figure 3.7, two renewal processes generate events with constant increments d_1 and d_2, respectively. The resulting process is not a renewal process.

Merging of renewal processes

Merging renewal processes will in general *not* result in a new renewal process.

3.1.3 Superposition of processes

The processes observed and used for analyzing in ICT-systems are very often an aggregate of many independent processes. For instance, the telephone calls initiated toward a destination is the aggregate of the individual calls from an almost infinite number of subscribers who may have an interest in calling this destination. If each of these processes is a renewal process with inter-event times $Y_{j,i}$, the aggregate process, $Y_{A,i}$ is a superposition of these, as illustrated in Figure 3.8.

In the above subsection, it was pointed out that the aggregate of renewal processes is not necessarily a renewal process. However, in the case a large number of processes are superposed, the aggregate process in most cases tends to a Poisson process, see Section 5.1. This is formalized in Palm-Khintchine's theorem.

Palm-Khintchine's theorem

The aggregate process resulting of the superposition of n independent renewal processes with inter-event times $Y_{j,i}, j = 1, \ldots, n$, such that $n/\sum_{j=1}^{n} E(Y_j) \to c$ where c is finite and $E(Y_j) \not\ll \sum_{j=1}^{n} E(Y_j)/n, \forall j$, i.e. no single process dominates the aggregate process, tends to a Poisson process when $n \to \infty$.

How fast the aggregate process approaches a Poisson process with increasing n depends, in addition to the number of subprocesses, on how close to a Poisson process each of these

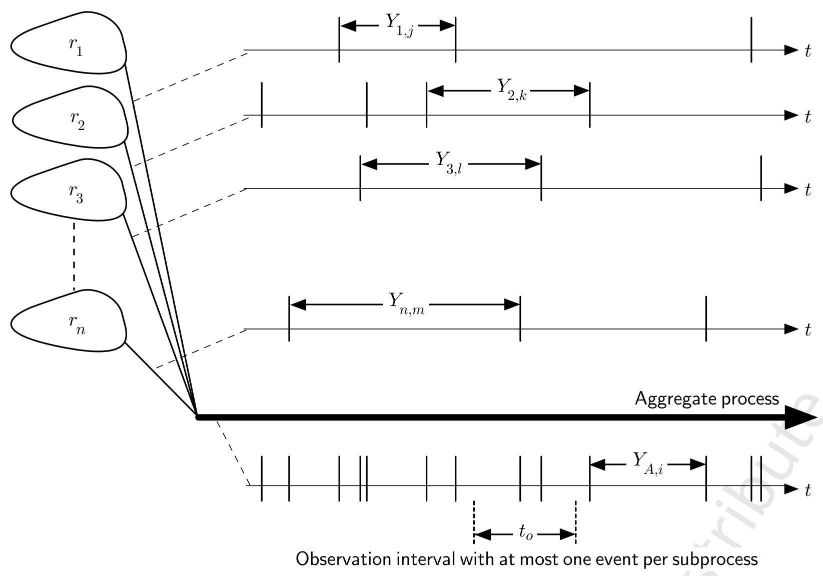


Figure 3.8: Superposition of a number of renewal processes

processes is. Note also that when the number of subprocesses is so large and the time the aggregated process is observed is so short that there is at most one event per subprocess, the observed aggregate process behaves as a Poisson process over the observation interval. For instance, in Figure 3.8, it may be assumed that the number of events occurring in the (randomly located) interval of length t_o is approximately Poisson with parameter $t_o / \sum_{j=1}^{n} E(Y_j)$ also when the individual processes are not renewal processes.

3.2 Stationary and transient regimes

For the sake of simplicity, we consider in this section only one-dimensional processes, but the definitions, discussions and results can be extended to multi-dimensional processes.

3.2.1 Stationary regime

The concept of stationarity is important in the context of analysis and dimensioning of ICT-systems. A time and event process is *stationary* if its statistical characteristics do not change over time. Real-world time and event processes in ICT-systems are never stationary if we refer strictly to the mathematical definition. However, there are periods over which these processes behave as if they were stationary for most practical purposes.

Let $X(t)$ be a discrete space continuous time process. The following also applies to continuous space discrete time processes if "=" is replaced by "$\leqslant$".

Let $P(X(t) = i)$ be the probability that the system is in state i at time t.

Stationary process

The process $X(t)$ is *stationary* if the probability to find the process in a given state is independent of the time.

$$P(X(t) = i) = P(X(t + \tau) = i),\ \forall t > 0;\ \forall \tau > 0 \tag{3.12}$$

In other words, if we observe a stationary process at random points of time $t \geqslant t_s$, the state probability distribution will always be the same.

$$P(X(t) = i) = p_i \tag{3.13}$$

where p_i denotes the steady state probability that the system is in state i.

This does not mean that the states of the process at time t_1 and $t_2 = t_1 + \tau$ are independent. Generally, $P(X(t_2) = x_2 \mid X(t_1) = x_1) \neq P(X(t_1) = x_1)$. But, for a stationary process, the dependence is only related to the time difference $t_2 - t_1$, not to the absolute values of t_1 and t_2.

Example 3.4 Stationary process
A continuously maintained telephone system can be assumed stationary with respect to whether it is operational or not[3]. That is, the probability that a call is successfully established is constant and independent of time. This probability is the asymptotic service availability

$$A(t) = P(\text{system is up at } t) = A$$

Now, if someone makes two call attempts within a short interval of time (e.g. 10 s) and the system is down at the first attempt, then the probability that the system is up when trying the second time is smaller than A.

$$P_{\text{OK}} = P(\text{system is up at } t + 10\ s \mid \text{system is down at } t) \leqslant A$$

However, P_{OK} is independent of t.

3.2.2 Transient regime

A process which is not stationary is said to be *transient*.

Time and event process always have a transient period before reaching a steady state which depends on the parameter values and dependences. Let χ be the dominating time constant that is the typical time between events or the typical time it takes to perform

[3]For a real system this is true under certain assumptions. Maintenance activities which may have negative effects on the failure intensity such as software upgrades and patches are conducted when the traffic is reduced, i.e. during the night. The failure intensity also depends on the traffic load which varies during the day and in the long-term.

the activities in the system which determines the duration of transient period, i.e. the time it takes for the process to reach a steady state.

A process can be transient mainly for two reasons:

- *The initial state distribution differs from the steady state distribution.* When a system is put into operation or when a simulation of a system is started, the initial state is known. For example when a system is put into operation ($t = 0$), the system is up and running and the instantaneous availability is $A(0) = 1$. Only after some time the system reaches a steady state where there is a certain probability that the system has failed. Another example is the simulation of a router. When the simulation starts the buffers are empty or filled up to a predefined level and only after some time the buffer length distributions stabilize (with constant traffic parameters).
- *The underlying parameters $\underline{\xi}(t)$ change over time.* Examples include: daily variations of the offered traffic illustrated in Figure 1.5 on page 17, software failure intensity decreases when faults are found and corrected. Upon modification of the underlying parameters, the system is not stationary but tends to be stationary if the parameters tends to become constant, i.e. $\lim_{t\to\infty} \underline{\xi}(t) = \underline{\xi}$.

  In practice, processes with underlying parameters changing over time can be studied as if they were stationary.

  - *Local stationarity.* The process is inspected over a long interval compared to the time constants and the parameters are approximated by constants.

    $$\underline{\xi}(t) \sim \underline{\xi},\ t \in [t_1, t_2] \text{ and } t_2 - t_1 \gg \chi \tag{3.14}$$

    The use of the busy hour, see page 34, is an example of such approach.

  - *Quasi-stationarity.* The underlying parameters change so slowly that the process can be considered stationary.

    $$\max\left(\frac{\frac{\mathrm{d}}{\mathrm{d}t}\underline{\xi}(t)}{\underline{\xi}(t)}\right) \ll \frac{1}{\chi} \tag{3.15}$$

    Quasi-stationarity can be used for example to compute the availability of a system over a period when dependability increases.

Example 3.5 Unreliable software
One of your classmates has installed an unreliable software on his/her PC. This software crashes on average after 30 minutes, and it takes your classmate on average 9 minutes to restart the PC, clean the disk and get the program up and running again. You have had enough of cursing and swearing and you want to know the probability that the software works if you leave your classmate when he/she has just got the software working and you come back after a time t.
Your classmate's PC (and mood) follows a stochastic process $X(t)$ continuous in time and with a discrete state space with two states, $\Omega = \{\text{Up}, \text{Down}\}$. $A(t) = P(X(t) = \text{Up})$ is the instantaneous

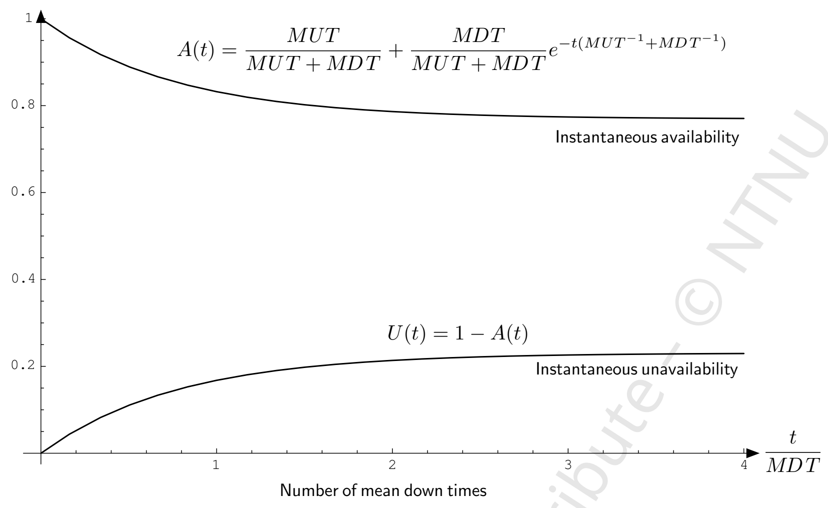


Figure 3.9: Availability for a system with $MUT/MDT = 10/3$

availability of the system. After having observed the process you think the time to crash and the time from crash to up state can both be approximated by negative exponential distributions

$$P(\text{uptime} > t) = e^{-\frac{t}{\text{MUT}}} = e^{-\frac{t}{30\text{ min}}}$$

$$P(\text{downtime} > t) = e^{-\frac{t}{\text{MDT}}} = e^{-\frac{t}{9\text{ min}}}$$

The result for the availability is shown in Figure 3.9.

If MUT is significantly larger than MDT, the time constant is dominated by MDT and after approximately $3 \cdot MDT$ the information that the software was just restarted when you left is "forgotten". Therefore if you want to use this information, you cannot be away for a long time, regardless how reliable the software is.

3.2.3 Variations of the intensity

We distinguish between several types of variations. The most important are:

- *Periodic variations*, e.g. daily variations of the offered traffic as illustrated in Figure 1.5 on page 17 (a packet arrival or a new call are the events for such a point process).
- *Trends* (long-term variations), e.g. increase of the traffic due to new services or increase of the dependability of the system after a fault removal. If a process is studied over a reasonably short period of time, trends are negligible. However, in some cases it is important to detect them, e.g. when studying the performance of a system when a new service is delivered or the increase of the failure intensity due to the wear and tear of physical components.

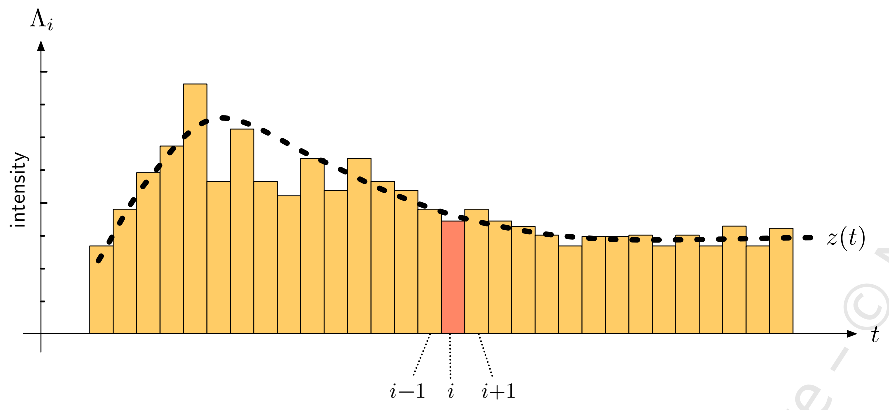


Figure 3.10: Number of observed events during an interval Δ as a function of time

- *Stochastic variations*, variations induced by changes in the environment affecting some underlying parameters and therefore hardly predictable, e.g. lightening during summer and the induced failures due to electrical overstress in the interface electronics of communication devices.

How much these variations affect the intensity of a process depends on how the intensity is measured. When observing a process over a given period of time, it is natural to divide the time in intervals and then study the number of events per interval, see Figure 3.10. The intensity observed is then the number of events divided by the interval length

$$\Lambda_i = \frac{N_i}{\Delta} \tag{3.16}$$

and we have

$$\frac{1}{\Delta}\int_{(i-1)\Delta}^{i\Delta} z(t)\,\mathrm{d}t = \mathrm{E}(\Lambda_i) \tag{3.17}$$

By letting Δ tend to 0, the intensity of the process can be observed at any time t. But the smaller Δ, the fewer the number of observations per interval and the larger the uncertainty on the estimation, see Chapter 8. In practice, Δ is therefore chosen "reasonably small".

3.3 Dependence and independence

As in the previous section, let $X(t)$ be a discrete space continuous time stochastic process where, for the sake of simplicity, $X(t)$ is a one-dimensional process. The following also applies to discrete time processes and continuous space processes if cumulative distribution functions are used instead (i.e. "$=$" is replaced by "$\leqslant$" in the following equations).

Statistical independence has been defined in 2.1.4. This definition also applies to time and event processes. That is, if $P(X(t) = i)$ denotes the probability that the system is in state i at time t, the process is independent if

$$P(X(t+\tau) = i \mid X(t) = j) = P(X(t+\tau) = i),\ \tau \neq 0,\ \forall(t, i, j) \tag{3.18}$$

A process where the state of the system at a time t neither depends on the state of the system at another time nor has an influence on the future of the process, is not particularly interesting when studying the dependability and performance of a system. All the processes used to analyze a system have therefore some kind of dependence.

3.3.1 Markov processes

Markov processes are an important class of processes which will be described in more details in Section 5.2 and are often used for dependability and performance analysis.

Let $t_1, t_2, \ldots, t_n, t_{n+1}$ be random points of time where $t_1 < t_2 < t_3 \ldots < t_{n+1}$.

Markov process

A process is a *Markov process* if and only if the future evolution of the process from time t_n only depends on the state at time t_n and not on any previous state, i.e.

$$\begin{aligned} &P(X(t_{n+1}) = x_{n+1} \mid X(t_n) = x_n,\ X(t_{n-1}) = x_{n-1},\ \ldots X(t_1) = x_1) \\ &= P(X(t_{n+1}) = x_{n+1} \mid X(t_n) = x_n) \end{aligned} \tag{3.19}$$

This *memoryless property* leads to great simplifications. A process generally neither knows how it has reached a given state nor how long it has sojourned in this state. For a Markov process, the current state sums up the entire past history.

Semi-Markov processes

A process is called *semi-Markovian* if (3.19) holds only at times of state changes, while it holds at all time epochs for a Markov process. A semi-Markov process “remembers” how long it has been in a state but not how it reached this state and its future evolution does not depend on the sojourn time in this state.

Renewal processes, as defined on page 91, are a class of semi-Markov processes. If $N(t)$ denotes the number of events until and including time $t \leqslant t_n$ then

$$\begin{aligned} &P(N(t) = n \mid N(t_n) = n_n,\ N(t_{n-1}) = n_{n-1},\ \ldots N(t_1) = n_1) \\ &= P(N(t) = n \mid N(t_n) = n_n) \end{aligned} \tag{3.20}$$

where $t_1, t_2, \ldots, t_n$ are renewal points.

3.3.2 Autocovariance and autocorrelation

In Section 2.3.6, the covariance and the correlation coefficient have been introduced as measures of dependence between random variables. The same measures are defined for time and event processes, but are then called the *autocovariance* and the *autocorrelation*, respectively. The prefix *auto* refers to the fact that we consider the covariance of a process against a time-shifted version of itself.

Autocovariance

Covariance of a process $X(t)$ against a time-shifted version of itself.

$$\begin{aligned}\sigma_{X(t)X(t+\tau)} &= \mathrm{Cov}(X(t), X(t+\tau)) \\ &= \mathrm{E}\left((X(t) - \mathrm{E}(X(t)))(X(t+\tau) - \mathrm{E}(X(t+\tau)))\right)\end{aligned} \tag{3.21}$$

Covariance stationarity

A process $X(t)$ is *covariance stationary*, or *weakly stationary*, if

- the second order moment is finite

$$\mathrm{E}(X(t)^2) < \infty \, , \tag{3.22}$$

- the expected value is finite and constant

$$\mathrm{E}(X(t)) = \mathrm{E}(X(t+\tau)) = \mu < \infty, \ \tau > 0 \, , \text{ and} \tag{3.23}$$

- the autocovariance is finite and only depends on the time difference $t_2 - t_1$

$$\sigma_{X(t_1)X(t_2)} = \sigma_{X(t_1+\tau)X(t_2+\tau)} \equiv \sigma_{XX}(t_2 - t_1) < \infty, \ \forall\tau, \ t_1, \ t_2. \tag{3.24}$$

From the definition, it follows that the autocovariance function of a covariance stationary process is even, i.e. $\sigma_{XX}(\tau) = \sigma_{XX}(-\tau)$, $\forall\tau$, and $\sigma_{XX}(0) = \sigma_X^2 = \mathrm{Var}(X(t))$.

If $\mathrm{E}(X(t)^2) < \infty$, then (strict) stationarity, as defined in Section 3.2.1, implies covariance stationarity.

Autocorrelation

The *autocorrelation* of a covariance stationary process is defined as

$$\rho_X(\tau) = \frac{\sigma_{XX}(\tau)}{\sigma_{XX}(0)} \tag{3.25}$$

From the definition, it follows that the autocorrelation is also an even function, i.e. $\rho_X(\tau) = \rho_X(-\tau)$.

The autocorrelation gives an indication of the dependence in the process which is to be interpreted in a similar way as the correlation in Section 2.3.6. See also Example 3.6.

Example 3.6 Unreliable software (continued)
Let us come back to Example 3.5 and find the autocorrelation of the process $X(t)$. Let $X(t) = 1$ when the software is up and running, 0 otherwise.
Then, for a stationary process, we have:

$$A = P(X(t) = 1) = \mathrm{E}(X(t)) = \mu = \frac{MUT}{MUT + MDT}$$

In addition, $\mathrm{E}(X(t)X(t+\tau))$ can be found from Figure 3.9.

$$\begin{aligned}\mathrm{E}(X(t)X(t+\tau)) &= \mathrm{E}(X(\tau) \cdot X(0)) \\ &= P(X(\tau) = 1 \cap X(0) = 1) \\ &= P(X(\tau) = 1 \mid X(0) = 1) \cdot P(X(0) = 1) \\ &= (A + (1-A)e^{-t(MUT^{-1} + MDT^{-1})}) \cdot A\end{aligned}$$

Using (3.21) and (3.25) we find

$$\rho_{\mathrm{Up-Down}}(\tau) = e^{-t(MUT^{-1} + MDT^{-1})}$$

That is, the autocorrelation decays exponentially with the time constant $(MUT^{-1} + MDT^{-1})^{-1}$.

Example 3.7 Failures in a telephone switch
Software failures in a telephone switch constitute a point process $N(t)$. It can be converted to a discrete space discrete time process by letting the state of the process be the number of software failures in 8 hours. Let $n(t_i) = N(t_i) - N(t_{i-1})$ be the number of failures between t_{i-1} and t_i, where $t_i - t_{i-1} = 8$ hours. For this discrete time process, the autocorrelation can be rewritten

$$\rho(j) = \frac{\mathrm{Cov}(n(t_i), n(t_{i-1}))}{\mathrm{Var}(n(t_i))} = \frac{\mathrm{E}(n(t_i), n(t_{i-1})) - \mathrm{E}(n(t_i))\mathrm{E}(n(t_{i-1}))}{E\,(n(t_i)^2) - \mathrm{E}(n(t_i))^2}$$

This process is observed for a telephone switch and the resulting autocorrelation is shown in Figure 3.11. The figure shows that there is a significant dependence between the failures that occur in the following time intervals. This contradicts the general assumptions that such failures occur independently. The two dashed lines indicate the 95% confidence bounds of the autocorrelation values if the failures were independent. The failure process "remembers" the failures during more than a week. Note also that the dependence in terms of number of failures is higher between consecutive days than consecutive 8 hour intervals.

Example 3.8 MPEG video
An MPEG video consists in a series of frames of different types. Frames in an MPEG video are arranged in a specified order called the GOP (Group of Pictures) pattern, e.g. a 12-frame sequence. See for instance [21] for a short introduction. The number of bytes in each GOP constitutes (approximately) a discrete time continuous space process. Since MPEG video is an important type of traffic load in the Internet, we are interested in knowing to which extent consecutive GOPs are dependent on each other. Figure 3.12 shows the autocorrelation for an example MPEG video, where the duration of a GOP is approximately 30 seconds.

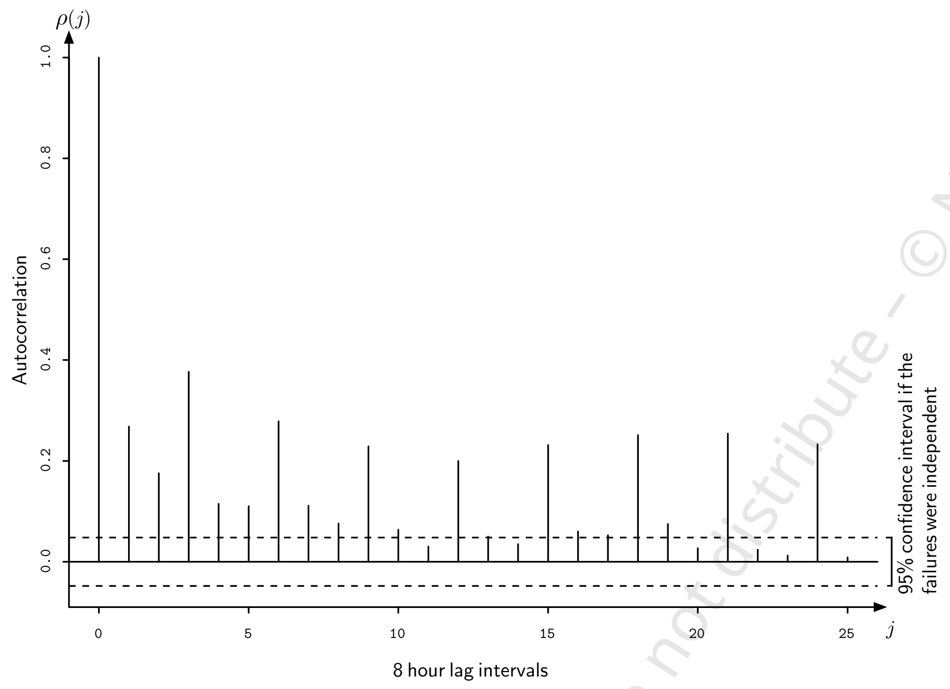


Figure 3.11: Autocorrelation $\rho(j)$ between software failures in a telephone switch

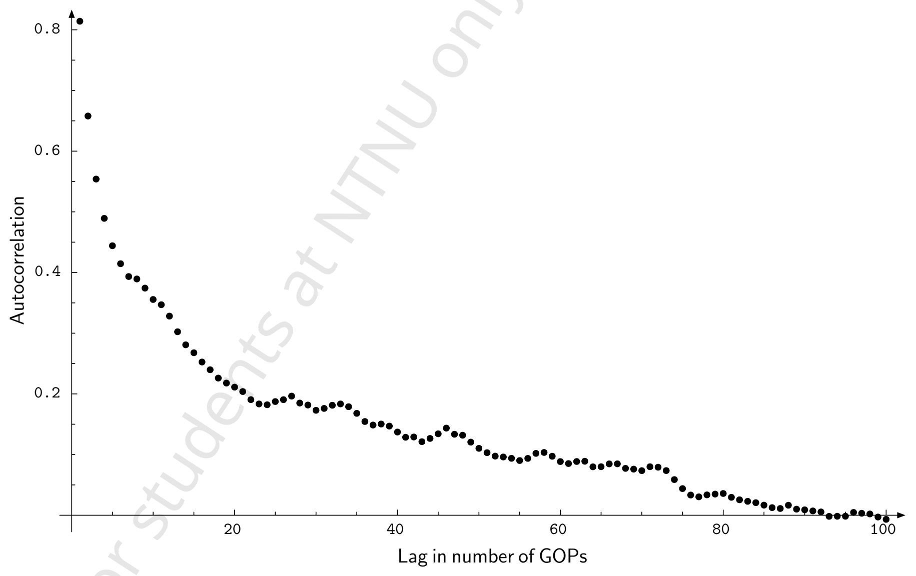


Figure 3.12: Autocorrelation in an Asterix cartoon [22]

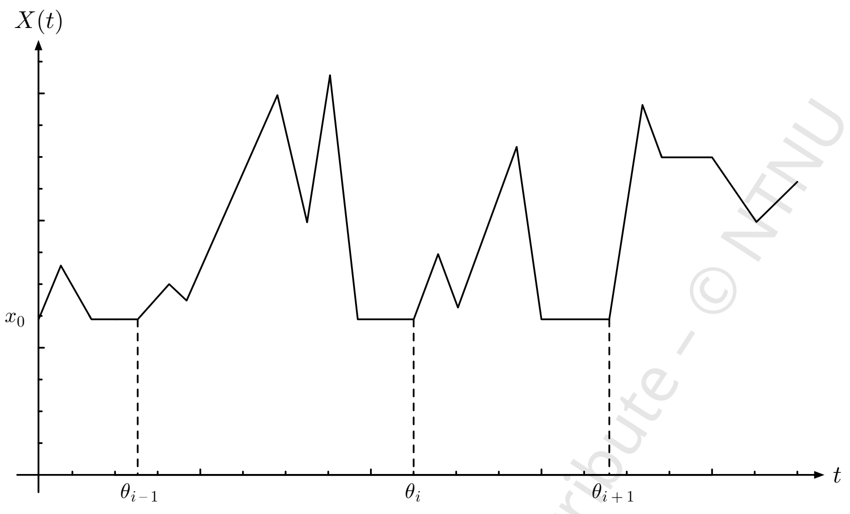


Figure 3.13: Regenerative process

3.3.3 Regenerative processes

A *regenerative process* is a stochastic process that has a number of points, independent of each others, at which it statistically "regenerates", i.e. at which its future evolution does not depend on its past history. The behavior of the process between two successive *regeneration points* is called a *regeneration cycle*. Such a process is illustrated in Figure 3.13, where $\{x_0, \theta_i\}$ are the regeneration points.

Regenerative process

The process X is *regenerative* and $\{x_0, \theta_i\}$ are *regeneration points* if and only if

$$P(X(\tau + \theta_i) = x \mid X(\theta_i) = x_0) = P(X(\tau + \theta_j) = x \mid X(\theta_j) = x_0),\ \forall \tau \geqslant 0,\ \forall (i,j) \tag{3.26}$$

Regeneration points are random events which form a renewal process, see Section 3.1.2, where the increments are independent and identically distributed.

$$P(\theta_{i+1} - \theta_i \leqslant y) = P(\theta_i - \theta_{i-1} \leqslant y) \tag{3.27}$$

All the properties related to the behavior of the process in different regeneration cycles are independent and identically distributed, e.g. the integral of the process

$$P\left(\int_{\theta_{i-1}}^{\theta_i} X(t)\,\mathrm{d}t \leqslant x\right) \tag{3.28}$$

Two important questions remain; which processes have regeneration points and how to find them?

- From the definition of a Markov process (3.19), a Markov process regenerates at all times and regeneration points are found in all states.
- For a semi-Markov process, the time epochs the process enters a state ($X(\theta_-) \neq x_0$ and $X(\theta_+) = x_0$) or leaves a state ($X(\theta_-) = x_0$ and $X(\theta_+) \neq x_0$) are two alternative regeneration points.
- Generally, there are possible regeneration points when a process has Markov properties. For a discrete space process, we have a regeneration point if we can find a state x_0 and a point of time in this state (exactly when entering the state or immediately before leaving it) where the future evolution of the process does not depend on its past history. Such states are for example when a system is "empty" or "failure free" and the external process that causes new events is a Poisson process, see Section 5.1, where the times between events are independent and identically negatively exponentially distributed.

If a process has several alternative regeneration points, it is in general advantageous to choose the most frequent one.

Example 3.9 Regeneration points for a server
Consider the system shown in Figure 3.4. If the job arrival process is a Poisson process (which will be the case if there is a large number of users (Palm-Khintchine's theorem)) then the system "forgets" its past history when it goes empty. The instant at which the system goes empty (or the next job arrives) is therefore a regeneration point.

Example 3.10 Regeneration points of a failure process
Assume that failures occur according to a Poisson process. If repair or any other restoration mechanism (e.g. restart) always brings the system back to a state as if it was new, then "just repaired" and "new failure" (i.e. either when the system enters or leaves the state where all its elements are as new) are two alternative regeneration points. The regeneration points are independent of the number and types of failures, even if the restoration mechanism itself is not memoryless.

Finally, note that the existence of regeneration points is based on assumptions on the properties of the process. This implies that we can base the analysis on the existence of such regeneration points only if we have full control on these assumptions. It is the case when:

- measurements are made on a simulated process,
- measurements are made on a real system where we have full control on the environment, e.g. a server artificially loaded, or
- mathematical analysis is done on a well-known process.

When carrying out measurements on a real system in an uncontrolled environment, the study cannot be based on regeneration points. For instance, in a dependability study, if it is assumed that the period between two failures of an intact system constitutes a

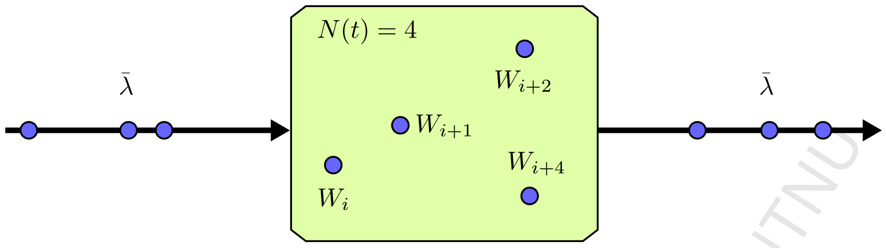


Figure 3.14: Illustration of a system with an average arrival intensity $\bar{\lambda}$ and a random sojourn time in the system W

regeneration cycle while the failures are actually correlated as in Figure 3.11, the results will be wrong.

3.4 Little's formula

Little's formula gives a fundamental relation between the average arrival intensity, the average sojourn time and the average number in the system and is one of the few general results for time and event processes.

Consider the system shown in Figure 3.14 which can be described as a stochastic process in time and space. Customers arrive with an average intensity $\bar{\lambda}$. The system is stationary and we are interested in the number of customers in the system. A customer can be a packet in a packet-switched system, a failure etc. There is a relation between the average arrival intensity, the average sojourn time and the average number of customers in the system.

Let

$N(t)$	number of customers in the system at time t
W_i	sojourn time in the system for customer i
$\bar{\lambda}$	average arrival intensity

Let h be the number of events (arrivals) over a long time period τ. From Figure 3.15 it is seen that

$$\int_0^\tau N(t)\,\mathrm{d}t = \sum_{i=1}^{h} W_i \tag{3.29}$$

In this example the system is empty both at time $t = 0$ and $t = \tau$. If it is not the case, (3.29) has additional terms. However, when τ is sufficiently large, the contribution of these terms is negligible so they can be disregarded.

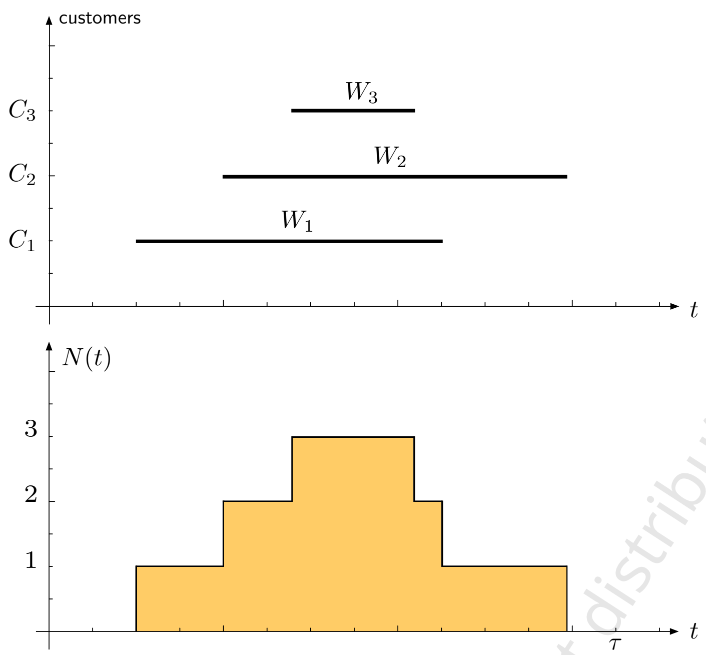


Figure 3.15: Relation between the number of customers in the system and the sojourn times

The arrival intensity over the interval $[0, \tau]$ is $\bar{\lambda} = h/\tau$ which gives

$$\frac{1}{\tau}\int_0^\tau N(t)\,\mathrm{d}t = \frac{1}{\tau}\sum_{i=1}^{h} W_i = \frac{\bar{\lambda}}{h}\sum_{i=1}^{h} W_i \qquad (3.30)$$

Little's formula

The average number of customers in a stationary system is equal to their average arrival intensity multiplied by their average sojourn time in the system.

$$\bar{N} = \bar{\lambda}\cdot\hat{W} \qquad (3.31)$$

Note that Little's formula applies for any arrival process and distribution of the service time in the system, and that (3.31) is exact when $\tau \to \infty$, i.e. $\mathrm{E}(N) = \mathrm{E}(\lambda)\cdot\mathrm{E}(W)$.

Example 3.11 Mean time students study at NTNU
The mean time students study at NTNU can be found if the number of students accepted each year and the mean number of students at NTNU are known.

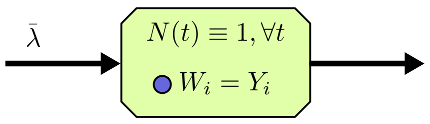


Figure 3.16: System where a customer replaces another

Example 3.12 Printer queue
A laser printer prints on average one job per 5 minutes, i.e. $\bar{\lambda} = 1/(5\,[\mathrm{min}])$. By sampling the printer queue during one day (which gives the number of jobs pending and the number of jobs served), it is found that the mean number of jobs in the system is $\bar{N} = 0.73$. Hence, it takes on average $\hat{W} = \bar{N}/\bar{\lambda} = 3.65\,[\mathrm{min}]$ to print a job.

Example 3.13 Holding time of a resource
In Section 1.2.3, A' has been defined as the mean number of busy resources (1.15). If the average arrival intensity is equal to $\bar{\lambda}$ then, using Little's formula, we have the following relation

$$A' = \bar{\lambda} \cdot \hat{T}_B$$

where $\hat{T}_B$ denotes the average holding time of a resource.

Example 3.14 Bandwidth-delay product
The bandwidth-delay product has been introduced on page 38. The throughput (bandwidth) is here the arrival intensity of bits, so the product gives the mean number of bits in transit in the network.

Example 3.15 Failure intensity
Assume that failures occur only when the system is working (up). If the average failure intensity is equal to $\bar{\lambda}$, then using Little's formula, the mean number of failures in the system is $\bar{\lambda} \cdot MDT$. And, since there is at most one failure at a time in the system, we have $U = \bar{\lambda} \cdot MDT = MDT/MTBF$, where U denotes the unavailability of the system, i.e. the probability that the system is failed. The relation $\bar{\lambda} = 1/MTBF$ is given by Korolyuk's theorem (see below). We have used Little's formula to derive a relation previously given (1.4) on page 29.

Little's formula can also be used to find the relation between the average time between events and the average intensity. Let the events be customers arriving to a system and stay in the system until the next arrival, see Figure 3.16. Therefore, there is always one and only one customer in the system and the mean sojourn time is equal to the mean inter-arrival time, $\hat{Y}$. Using Little's formula,

Korolyuk's theorem

$$\bar{\lambda} = \frac{1}{\hat{Y}} \tag{3.32}$$

Note that (3.32) is identical to the result given for renewal processes (3.9). Since Little's formula is independent of the arrival process, this is also the case of Korolyuk's theorem.

4 Simulation

As mentioned in Chapter 1, three different methods can be used to analyze and dimension systems with respect to non-functional requirements, namely mathematical analysis, simulation and measurements on a prototype. This chapter focuses on simulation and how it can be used.

Simulation

Imitative representation of the functioning of one system or process by means of the functioning of another. *[Webster]*

A simulation, in the context of this book, is performed by means of a software program, called a *simulator*, running on a computer. A simulator is an instance of a *simulation model* which includes the elements presented in Figure 1.2. A simulation model describes all the components that affect the properties to evaluate, as well as structural components and how they work together. The environments of the system that influence these properties should also be part of the model, e.g. the traffic (number of packets per second, number of calls, call duration), the failure process (time to failure, failure rate, failure propagation), the maintenance (time to repair).

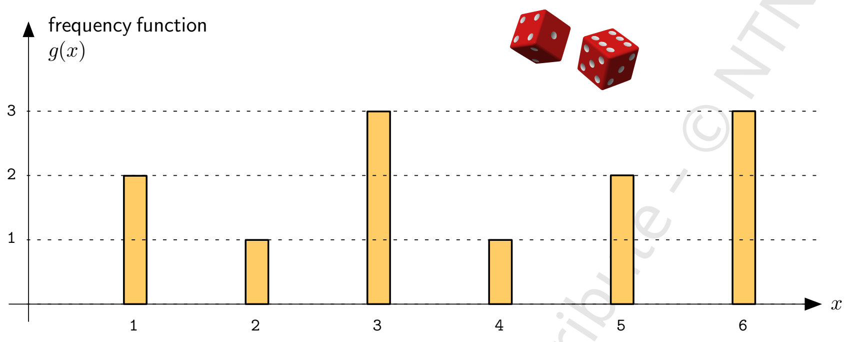
In this chapter, by simulation we refer to stochastic simulation, i.e. the behavior of the system components is, fully or partly, governed by stochastic processes. To imitate a stochastic behavior, the simulator, and therefore the computer, has to be able to generate random variables according to a given probability distribution.

This chapter is organized as follows. First, Section 4.1 discusses the generation of random variables. Discrete-event simulation is then introduced in Section 4.2. For supporting literature, see for instance [6, 7] and [23]. For further details about process-oriented simulation and in particular *Discrete Event Modelling on Simula (DEMOS)*, see [24, 5].

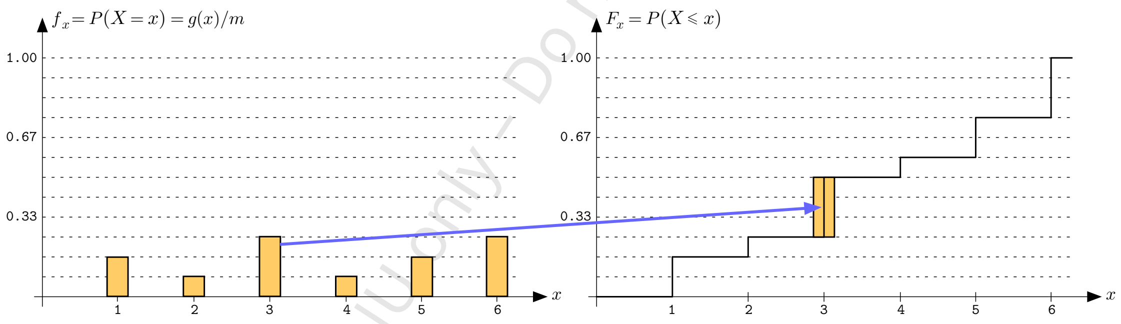
4.1 Random variate generation

As mentioned in the introduction, simulation relies on the generation of random variables and therefore on the generation of random numbers. Generated random variables are called *random variates*. To understand how random numbers can be used to generate random variates according to a given distribution, consider the following example.

Example 4.1 Dice rolling
A dice is rolled several times and each time the number obtained between 1 and 6 is noted. In the figure below, the same dice has been rolled 12 times and the observations are assumed to be representative of this particular dice.

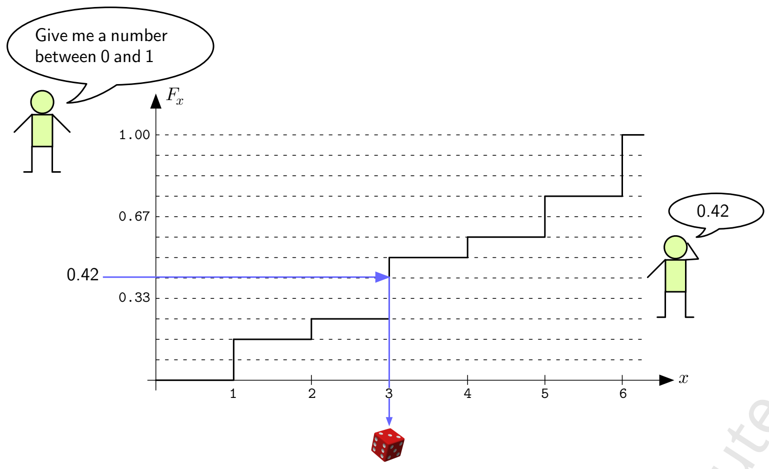


The y-axis is rescaled to convert the frequency function $g(x)$ to the probability mass function f_x, i.e. the number of times m_i the value $i \in \Omega = \{1, 2, \ldots, 6\}$ has been observed is divided by the total number of observations, here $m = 12$. The corresponding cumulative distribution function is F_x.



A method for drawing a random number from the distribution F_x consists in first drawing an uniformly distributed random number y between 0 and 1 and then reading the corresponding number on the x-axis, i.e. the value of x such that the cumulative probability is less than y (inverse transform technique, see Section 4.1.2). Here, $y = 0.42$ and $F_x \leqslant 0.42$ gives $x = 3$.

This example motivates the need to generate sequences of uniformly distributed random numbers between 0 and 1. Such sequences can be used to generate random variates from any distribution. Section 4.1.1 described how to generate independent and identically uniformly distributed random numbers and Section 4.1.2 how to draw random variates from any distribution.



4.1.1 Random number generation

Random numbers may be generated either from a physical process or from an algorithm. The main drawback of generating random numbers from a physical process is that the sequence is not reproducible. This can be worked around by recording the sequence, but this implies to store a large amount of data and it is not possible to record all types of physical processes. On the other hand, generating random numbers from an algorithm using a computer allows to generate a large number of numbers and to reproduce the same sequence several times which enables to reproduce the same experiments. Random number generated by an algorithm are not truly random since the algorithm itself is deterministic and are therefore called *pseudo-random numbers*. However, the sequence of numbers generated should appear to be *statistically random* (independent and uniformly distributed random numbers).

Generating a sequence of random numbers with a computer amounts to creating a sequence of integers according to an algorithm where an integer depends on one or more integers previously generated. For instance, let R_i be the i^{th} integer in a sequence. Assume that R_i is calculated as function g of the previous random integer R_{i-1}

$$R_i = g(R_{i-1}),\ i \in \mathbb{N}^* \tag{4.1}$$

The sequence of integers generated only depends on the first integer used R_0 (*seed*). Now, the number of different integers which can be generated by a computer is finite and equal to m (m depends on the number of bits used to represent integers, $m = 2^{<\#\text{bits}>}$). Therefore the sequence of integers produced by a computer recurs and the largest cycle length p (period) is equal to m. Formally,

$$R_{i+p} = R_i \text{ and } R_i \neq R_{i+j},\ \forall i \in \mathbb{N},\ j \in \{1, 2, \ldots, p-1\} \tag{4.2}$$

If for any sequence $\{R_1, R_2, \ldots\}$ where $0 \leqslant R_i \leqslant m-1$, the R_i are independent and uniformly distributed, then R_i are (pseudo-)random numbers.

As mentioned in the introduction of this section, the goal of any generation process is to produce sequence of numbers between 0 and 1 that appear independent and uniformly distributed. Generating uniformly distributed continuous random variates $U \in (0, 1]$ is approached by generating uniformly distributed integers $R \in \{0, 1, 2, \dots, m-1\}$ and computing R/m. $U_i = R_i/m$ are independent and uniformly distributed over $(0, 1]$.

Requirements for a random number generator

Before detailing a method to generate random numbers, requirements for any random number generator are listed.

- *Fast.* The random number generation routine should be fast all the more so as simulation may require many random numbers.
- *Portable.* The algorithm should be portable to different programming languages and different computer architectures.
- *Long cycle.* The cycle length, or period, $p \leqslant m$ is the number of random numbers in a sequence before it repeats. The cycle length should be as long as possible. A *full period generator* has a period p equal to $m-1$. For instance, in the case of a multiplicative congruential generator (see below), 0 cannot be used and a full period generator has a period of $m-1$.
- *Reproducible.* Given a seed, it should be possible to reproduce the same sequence of random numbers. In a simulation study, this enables to reproduce the same experiment.
- *Good statistical properties.* The generated random numbers U_i should appear independent and identically uniformly distributed between 0 and 1.

Linear congruential method

There are a number of common methods used for generating uniformly distributed random numbers. The simplest and most widely used is the *linear congruential method.*

Let R_i be uniformly distributed, $R_i \in \{0, 1, 2, \dots, m-1\}$, and a and c be two constants. Random numbers are generated using the linear recurrence relation[1]

$$R_{i+1} = (aR_i + c) \bmod m, \ \forall i \in \mathbb{N} \tag{4.3}$$

The parameters R_0 (seed), a (multiplier), c (increment) and m (modulus) affect the statistical properties of the generated sequence and the cycle length. If $c \neq 0$ the method is called *mixed congruential method.* If $c = 0$ the method is known as the *multiplicative congruential method.* Note that in this case the seed R_0 must be chosen non-null.

[1]Modulo operation: $a \bmod b = a - b\lfloor \frac{a}{b} \rfloor$, where $\lfloor x \rfloor$ is the floor function, i.e. $\lfloor x \rfloor$ is the largest integer less than or equal to x.

The cycle length p should be the largest possible. The optimal period can be achieved by a proper choice of parameters [6, 25].

- For $m = 2^b$ and $c \neq 0$, the longest period $p = m = 2^b$ is achieved if c is relatively prime to m [2] and $a = 1 + 4k$ where k is an integer, $k \in \mathbb{Z}$.
- For $m = 2^b$ and $c = 0$, the longest period $p = m/4 = 2^{b-2}$ is achieved provided that R_0 is odd and $a = 3 + 8k$ or $a = 5 + 8k$ where $k \in \mathbb{N}$.
- For m a prime number and $c = 0$, the longest period is $p = m - 1$ and is achieved if a is not divisible by m for $k < m - 1$ where k is an integer.

Example 4.2 Random number generator in DEMOS
The generator used in DEMOS is a Lehmer generator [26].

$$R_{i+1} = (8192 R_i) \bmod (67099547)$$

This is a multiplicative congruential generator where $c = 0$, $R_0 = 33427485$ (standard value, but it can be modified by the user), $a = 8192$ and $m = 67099547$. Theses values are chosen so that there is no overflow on a 32-bit computer.
The generator is implemented in Simula as follows [5]:

```
FOR K := 32, 32, 8 DO
  BEGIN
    U := K*U;
    IF U >= 67099547 THEN U := U - U//67099547*67099547;
  END;
```

Tests for random numbers

Departures from the desirable properties of pseudo-random numbers U_i (real randomness, uniformity and independence) can be tested[3].

- *Non-uniform distribution.* Compare the distribution of the set of random numbers generated to a uniform distribution e.g. using the Kolmogorov-Smirnov test (*frequency test*).
- *Discrete instead of continuous values.* Using the linear congruential method the R_i are always discrete and so are therefore the $U_i = R_i/m$. However, for large values of m, the sample space for U_i is approximately continuous over $(0, 1]$.
- *Mean too high or too low.* Compare the actual value of the mean to the theoretical expected value, see Example 4.3.
- *Variance too high or too low.* Compare the actual value of the variance to the theoretical variance, see Example 4.3.

[2]Two numbers a and b are *relatively prime* if the greatest common factor of a and b is 1.
[3]Tests for the uniformity, the mean and variance are only relevant for non-full period generators or a partial sequence of a full length generator.

- *Cyclic variations.* Several tests can be used including:
  - *Run test.* Compare the number of times the generated number is larger than the previous one and the number times it is smaller. Using the chi-square test to test if there is increasing of decreasing trend in the sequence or the partial sequence.
  - *Autocorrelation test.* Test the correlation (3.25) between the generated numbers, i.e. quantify the dependence between the numbers.
  - *Gap test.* Count the number of digits between particular digit patterns.
  - *Poker test.* Compare the frequency of numbers combinations to the expected values using the chi-square test.

Example 4.3 Expected value and variance of U
The uniformly distributed variates $U \in (0,1]$ are computed as $U = R/m$ where $R \in (0,m]$ are random numbers generated by a full period generator and m large.
The probability mass function of R is

$$f_r = \frac{1}{m}, \; r = 1, 2, \ldots, m$$

The expected value of U is

$$\mathrm{E}(U) = \frac{1}{m}\sum_{r=1}^{m} r f_r = \frac{1}{m^2}\sum_{r=1}^{m} r = \frac{1+1/m}{2}$$

The variance of U is

$$\begin{aligned}\mathrm{Var}(U) = \mathrm{E}\left(U^2\right) - \mathrm{E}(U)^2 &= \frac{1}{m^2}\sum_{r=1}^{m} r^2 f_r - \mathrm{E}(U)^2 \\ &= \frac{1}{m^3}\sum_{r=1}^{m} r^2 - \mathrm{E}(U)^2 \\ &= \frac{(2m+1)(m+1)}{6m^2} - \frac{(m+1)^2}{4m^2}\end{aligned}$$

Ideally U should be uniformly distributed over (0,1], $U^* \sim \mathcal{U}(0,1)$. From Table 2.1, we have

$$\mathrm{E}(U^*) = \frac{1}{2} \text{ and } \mathrm{Var}(U^*) = \frac{1}{12}$$

When $m \to \infty$

$$\lim_{m\to\infty} \mathrm{E}(U) = \frac{1}{2} = \mathrm{E}(U^*) \text{ and } \lim_{m\to\infty} \mathrm{Var}(U) = \frac{1}{12} = \mathrm{Var}(U^*)$$

4.1.2 Techniques for generating random variates

This section shows how random numbers are used to generate random variates from any probability distribution. Namely, this section discusses successively the *direct method*, the *inverse transformation technique*, the *acceptance-rejection technique* and the convolution method.

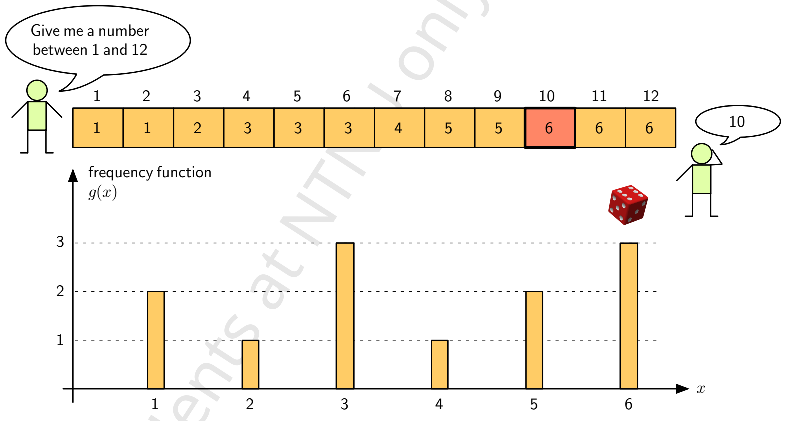
Direct method

The *direct method* is a simple and effective method for generating independent and identically distributed random variates from an empirical discrete probability distribution. It is similar to drawing from a hat with replacement. Assume that the hat contains a set H of values obtained empirically and that the cardinality of H is $|H| = m$ (number of observations). The sample space is denoted Ω and $|\Omega| = n$. One or more values in H may be equal and all the values in Ω may not be present in H. If m_x denotes the number of instances of the value x in H, then $m = \sum_{i=1}^{n} m_i$ and $0 \leqslant m_x \leqslant m$, $\forall x \in \Omega$. The probability p_x that the value x is drawn is given by m_x/m.

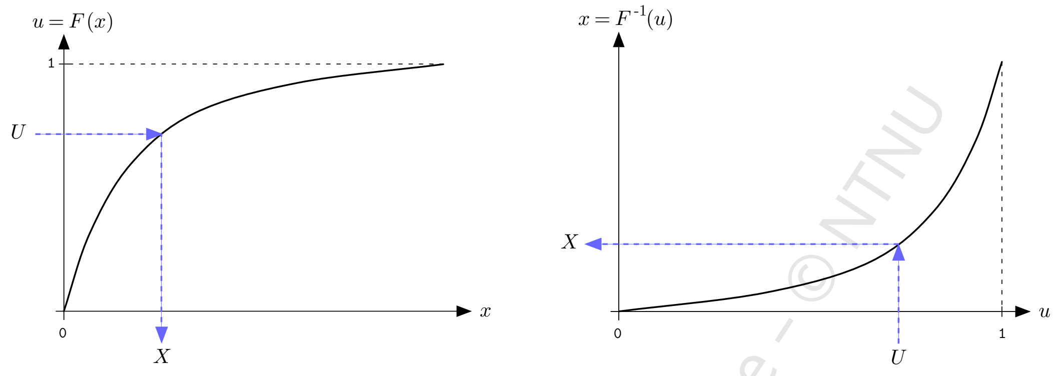
The number of elements m in H must be kept as low as possible since all these numbers must be stored in the computer memory and the random number generation routine should be fast. On the other hand, a precise representation of the probability distribution is desirable, i.e. $p_x = m_x/m$ should have a large number of significant digits, therefore m should be large enough. For large m, the cardinality of H may be reduced using the greatest common divisor between all the m_x.

H is organized as a (ordered or unordered) table. Generating a random variate X using the direct method simply consists in drawing a uniformly distributed Y between 1 and m and directly looking up the corresponding value for X in the table, $X = H[Y]$.

Example 4.4 Dice rolling and direct method
The dice rolling example (Example 4.1) illustrates perfectly the direct method. The $m = 12$ observations are ordered in vector $H = 1, 1, 2, 3, 3, 3, 4, 5, 5, 6, 6, 6$.



To generate a variate X, one draws an integer Y between 1 and m and reads the corresponding value of $X = H[Y]$. Here $Y = 10$ and $X = H[10] = 6$.



(a) Original distribution function **(b) Inverse distribution function**

Figure 4.1: Inverse transform technique

Inverse transform technique

The inverse transform technique can be used to generate a random variate X from a known cumulative distribution function $F(x)$ for which the inverse function $F^{-1}(x)$ can be calculated. It can be utilized for both continuous and discrete distribution, parametric as well as empirical. In the following, parametric distributions are discussed. For further details about the inverse transform technique and its variants, refer to [7].

Let X be a continuous random variable with probability density function $f(x)$ and cumulative distribution function $F(x)$. Since X is a random variable, $F(X)$ is a random variable uniformly distributed over $[0, 1]$, for a proof see for instance [23]. Let $U = F(X)$. By taking the inverse function F^{-1} of both sides of the equality we have

$$F^{-1}(U) = F^{-1}(F(X)) = X \tag{4.4}$$

Hence, if $F^{-1}(x)$ can be calculated, generating random variates $X \sim f(x)$ using the inverse transform technique consists in drawing uniformly distributed random numbers between 0 and 1 and calculating $X = F^{-1}(U)$. Figure 4.1 gives a graphical illustration of the technique.

Example 4.5 Inverse function for the exponential distribution
Let $X \sim \text{n.e.d.}(\lambda)$. From Table 2.1

$$F(x) = 1 - e^{-\lambda x}$$

Solving the equation $U = F(X)$ with respect to U gives:

$$1 - e^{-\lambda X} = U$$

$$e^{-\lambda X} = 1 - U$$

$$-\lambda X = \ln(1-U)$$

$$X = \frac{-\ln(1-U)}{\lambda}$$

Now $1-U$ can be replaced by U since both are uniformly distributed on over $[0,1]$ which yields to:

$$X = F^{-1}(U) = \frac{-\ln(U)}{\lambda}$$

Example 4.6 Inverse function for the uniform distribution
Let $X \sim \mathcal{U}(a,b)$. From Table 2.1

$$F(x) = \frac{x-a}{b-a}$$

Solving the equation $U = F(X)$ in terms of U gives:

$$\frac{X-a}{b-a} = U$$

$$X = a + (b-a)U$$

As already mentioned, the inverse transform technique can also be used to generate discrete random variates. Let X be a discrete random variable defined over $\Omega = 0, 1, \ldots, n$. The probability mass function of X is defined as

$$f(x) = P(X = x),\ \forall x \in \Omega \text{ and } \sum_{i=0}^{n} f(i) = 1 \tag{4.5}$$

and the cumulative distribution function as

$$F(x) = P(X \leqslant x) = \sum_{i=0}^{x} f(i),\ \forall x \in \Omega \tag{4.6}$$

To generate a random variate distributed according to F_X, the interval $[0,1]$ is divided in n sub-intervals of length $f(0), f(1), \ldots, f(n)$, respectively. Using the inverse transform technique consists then in drawing a uniformly distributed random number between 0 and 1 and finding to which interval U belongs to. See Figure 4.2 for an illustration.

The order in which the probability $f(i)$'s are used to partition the interval $[0,1]$ is indifferent. However, the order should be chosen to minimize the number of tests, e.g. the largest probabilities should be used first in case of a recursive search starting from 0. For further details see [25].

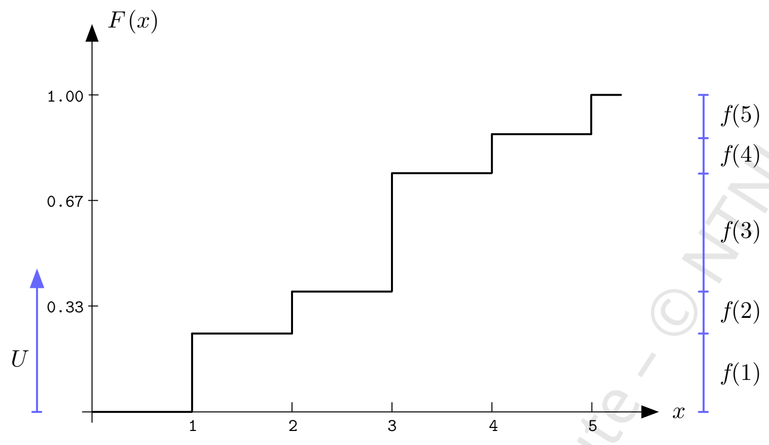


Figure 4.2: Cumulative distribution function for a discrete variable

When using the direct method, a uniformly distributed random number between 1 and m is drawn and used to look up a table H containing at most n different values. The probability of drawing the value x is $= m_x/m$, where m_x denotes the number of instances of x in H. If this value is used to estimate the point probability $f(x)$, the inverse transform technique described above can be used instead.

Acceptance-rejection technique

When the inverse cumulative distribution function F^{-1} cannot be written in closed form, e.g. for the normal distribution, the inverse transform technique cannot be used. The acceptance-rejection technique, or rejection sampling, is based on the principles of Monte-Carlo simulation and only requires the probability density function $f(x)$. It was first described in [27].

In its simplest form, this method requires that $f(x) \geqslant 0$ is defined over a finite interval $[a, b]$ and continuous. Let $c = \max\{f(x) \mid a \leqslant x \leqslant b\}$. The algorithm is as follows:

1. Generate a random variate X uniformly distributed over $[a, b]$.
2. Generate a random variate U uniformly distributed over $[0, c]$.
3. If $U \leqslant f(X)$ accept X, otherwise reject X and go back to 1.

Figure 4.3 illustrates this procedure. If the point of coordinates (X_i, U_i) is below the curve, X_i is accepted (smiling face), otherwise it is rejected (frowning face). Since U is uniformly distributed between 0 and c, a specific random variate X_i is accepted in $f(X_i)/c$ of the cases. Hence, since the X's are uniformly distributed over $[a, b]$, the accepted random variates are distributed according to $f(x)$.

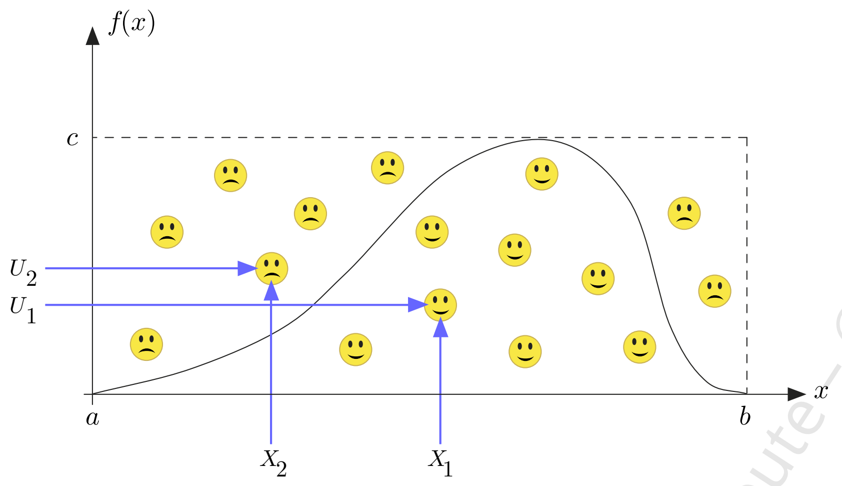


Figure 4.3: Acceptance-rejection technique

The efficiency of the acceptance-rejection technique heavily depends on the rejection ratio. If the rejection area is large, the technique is inefficient since on average a large number of drawings are needed to generate a random number distributed according to $f(x)$.

Assume now that we have a probability density function, $g(x)$, defined over the interval $[a, b]$ and from which we can easily draw random variates, e.g. using the inverse transform technique. To generate a random variate X with distribution $f(x)$, the technique consists then in drawing X from $g(x)$ and accepting it with a probability proportional to $f(X)/g(X)$. More precisely, the algorithm presented before is modified as follows:

1. Generate a random variate X from $g(x)$.
2. Generate a random variate U uniformly distributed over $[0, \alpha g(X)]$, where $\alpha > 1$ denotes a constant such that $\alpha g(x) \geqslant f(x)$, $\forall x \in [a, b]$.
3. If $U \leqslant f(X)$ accept X, otherwise reject X and go back to 1.

Note that this is equivalent to generate U uniformly distributed over $[0, 1]$ and test that $f(X)/(\alpha g(X)) \geqslant U$.

The closer the shape of $g(x)$ is to $f(x)$, the smaller the rejection area, and therefore the rejection ratio, is and the more efficient the technique is.

Convolution method

The probability density function of a sum of k independent random variables is equal to the k-convolution of the probability density functions of the random variables. The convolution method consists in generating and summing k independent random variates to generate a random variate. This method is illustrated in Example 4.7.

Table 4.1: Comparison of techniques for generating random variates

Method	Pros	Cons
Direct	Fast	Only applicable to discrete distributions
Inverse transform	Fast	Requires a closed form inverse cumulative distribution function
Acceptance-rejection	Many distributions	Requires several drawing per random variate
Convolution	Simple implementation for complicated distributions	Only applicable to sums of variates

Example 4.7 Generation of an Erlang-k distributed random variate
An Erlang-k distributed random variate can be generated by generating and summing k identically negatively exponentially distributed variates using the convolution method.
Let $X_i \sim \text{n.e.d.}(\lambda)$ and $Y = \sum_{i=1}^{k} X_i \sim \text{Erlang}(k, \lambda)$. The generator is

$$Y = \sum_{i=1}^{k} X_i = \sum_{i=1}^{k} -\frac{1}{\lambda} \ln U_i = -\frac{1}{\lambda} \sum_{i=1}^{k} \ln U_i = -\frac{1}{\lambda} \ln \left(\prod_{i=1}^{k} U_i \right)$$

Note that it is computationally more efficient to multiply the random variates generated first and compute only one logarithm.

Table 4.1 gives a brief comparison of the four methods presented in this section.

4.2 Discrete-event simulation

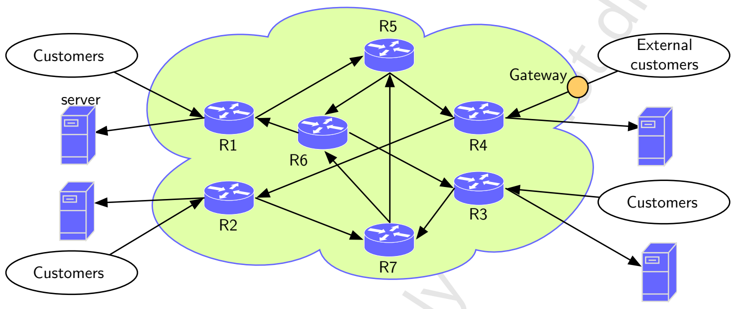
This section introduces simulation of discrete-event ICT-systems. A discrete-event system is a system which behavior is governed by events occurring at discrete points in time and resulting in distinct changes of the state of the system. A router with input and output buffers or a computer with software and hardware components are examples of such systems. Examples of discrete events for these systems are packet arrivals/departures and components failures/repairs, respectively. In a simulation, such events are emulated.

4.2.1 Introduction to simulation

Advantages and disadvantages of simulation

Simulation is a flexible and practical way of analyzing systems. Simulation allows to make models that include exactly the necessary level of granularity, not more, not less. A simulation model is better tailored for a given study than a mathematical model or a real system. A mathematical model often needs to be prohibitively simplified to be solved analytically. A real system used for a measurement study can obviously not be simplified at all. But for this very same reason, making a simulation model is also often more challenging and more demanding with respect to the insight into the system and the understanding of its operation. One must be able to remove unnecessary details and at the same time understand and describe the details of importance for a given study. Hence, a common and important by-product of a simulation study is an improved insight into the system and the identification of possible weaknesses in the design.

Example 4.8 Performance bottleneck
An Internet service provider wants to determine the bottleneck link of his network. It provides the topology of his network and explains that the server can be accessed from four routers, $R1, \ldots, R4$. His customers are connected to $R1$, $R2$ and $R3$ and external customers get access through $R4$. Every router has two output links and the load is balanced between the two links (50% on each) regardless the packet destination. A packet may therefore wander in the network before it reaches the server. What information do you need to be able to build a simulation model and solve this problem?



Application areas

In this chapter, simulation of ICT-systems is presented but simulation can be used in a wide range of application areas including (but not limited to):

- *Communication systems.* Routing strategies, resource management, maintenance policies, fault-tolerant systems.
- *Computer.* Performance of high performance computers or clusters.
- *Information systems.* Client/server architectures.
- *Verification of electronic circuits.* Function, logic.
- *Wave propagation in a transmission channel.* Propagation time.
- *Manufacturing systems.* Spare parts stocks, just-in-time production, manual vs. automated work.
- *Public systems.* Military (battle simulation, tactical maneuvers), health (medical waiting list), natural resources (waste management, oil spill)

- *Transportation systems.* Logistics, routing.
- *Construction systems.* Building, scheduling.
- *Restaurants.* Traffic in a fast-food restaurant.

Types of simulation models

Simulation models can be classified as static or dynamic, discrete or continuous, and deterministic or stochastic.

- A *static* simulation model (Monte-Carlo simulation) represents the system at a particular point of time, e.g. rolling a dice. A *dynamic* simulation model, on the other hand, represents a system as it changes over time, e.g. a set of users moving through a GSM cell and making calls.
- In a *discrete* simulation model, variables change only at discrete points in time (events). A *continuous* simulation model is one in which the variables vary continuously in time, e.g. the propagation of an electromagnetic wave in space.
- A *deterministic* simulation model does not contain any random variable. A given set of inputs result in a unique set of outputs, e.g. a "what-if" budget analysis in a spreadsheet. A *stochastic* simulation model contains one or more random variables as inputs which lead to random outputs. The output must therefore be treated as statistical estimates of the system characteristics.

In this book, we consider *dynamic discrete stochastic* simulation models.

Steps in a simulation study

A simulation study can be decomposed in several steps from the formulation of the problem to the conclusion. These steps are not isolated from each other, but connected as shown in Figure 4.4, and it may be necessary to go through some of them more than once. Note that the principles described below apply to mathematical analysis as well.

- *Problem formulation and objectives.* First, a problem should be identified. Then, the objectives of the study should be specified in collaboration with those who are familiar with the system, its requirements and the identified problem. An overall plan for reaching these objectives should also be established at this stage.
- *Model conceptualization.* All the components of the system that affect the properties to study must be included in the model, see Figure 1.2. This requires to be able to identify and understand the relevant details of the system for a given study and at the same time to abstract and disregard the irrelevant details. To this respect, modeling is as much an art as a science. It is impossible to give a general recipe for how to make a good simulation model.

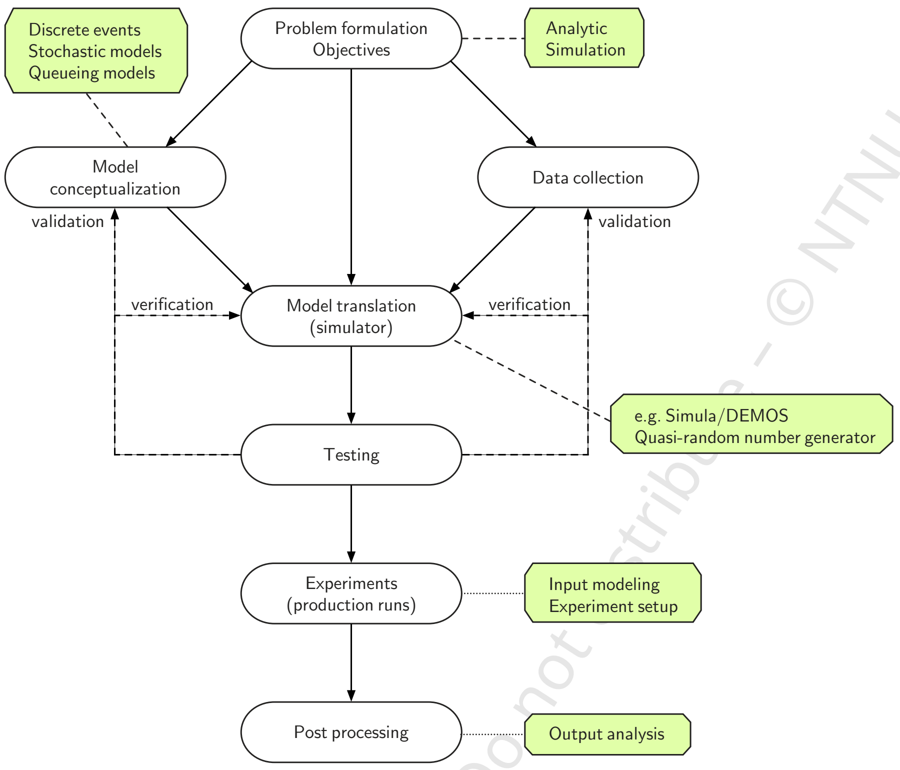


Figure 4.4: Flow chart of a simulation study

- *Data collection.* Similarly to the components of the system, information must be gathered about the behavior of the environment as illustrated in Figure 1.2. Typical parameters for the resources in the system, e.g. the probability distributions, must also be found. This is often a complex and time-consuming task. A very detailed description of the system is not of much help if no information about for instance the traffic, the failure rates, the typical buffer filling level is available. In such case, the results of the study are meaningless and no conclusion can be drawn[4].
- *Model translation.* Based on the abstract representation of the system, a simulation model must be designed. This model can then be implemented using simulation tools ranging from general programming languages to software tools specifically designed for a specific type of simulation models and systems, see Section 4.2.3.
- *Testing.* Once the simulator has been developed it must tested. That is, one must ensure that the simulator operates not only according to the model description (verification) but also imitate the real system that is modeled (validation). Depending on the output of the tests, it may be necessary to step back to the model

[4]The aphorism GIGO (Garbage In Garbage Out) is sometimes used.

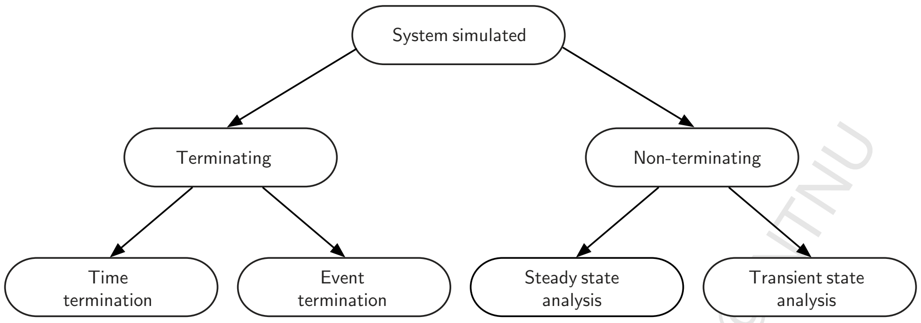


Figure 4.5: Types of simulation experiments

definition and correct errors in the simulator implementation before going further to the experiments.

- *Experiments.* In a simulation experiment, it may be desirable to run a single experiment with a fixed set of parameters during a long period of time and/or to run a large number of experiments with different parameters to study how sensitive the system behavior is to these parameters. Such production runs are usually time consuming and therefore must be thoroughly designed and planned. It is also recommended to have a procedure for handling the outputs of the simulation runs and log the progress of the experiment (parameters setting, simulation times, findings, etc.).
- *Post-processing.* After the experiments are (partially) completed, the outputs must be parsed (e.g. calculating statistics and values of interest, plotting graphs, etc.), analyzed and interpreted. Depending on the results obtained, additional experiments with new parameters may then be designed and run. Finally, results must be presented (e.g. writing of a report).

Types of simulation experiments

Simulation experiments can be classified according to the type of system simulated and to the type of properties studied, see Figure 4.5. Systems simulated can be classified as either *terminating* or *non-terminating*.

A terminating system stops after a certain number of events have occurred or after a given time has elapsed. For instance, a football match in a cup is typically a terminating system; the match terminates after 90 minutes, or, in case of a draw, after two extra time periods, or, if there is still a draw, after the penalty shootout. For terminating systems the initial conditions generally affect the measures. The length of a terminating simulation is determined by the system itself.

A non-terminating system does not naturally terminate. ICT-systems are typically non-terminating systems. When simulating a non-terminating system, initial conditions (system state) and termination conditions must be specified. Both transient-state analysis (e.g. time to failure, to repair, to service, to buffer overflow etc.) and steady-state analysis (e.g. mean traffic load for a router, a link, a switch, response time, call congestion, mean down time (MDT), mean up time (MUT), mean idle time etc.) are interesting.

For transient-state analysis, terminating conditions are specified as a number of events or a time duration. For steady-state analysis, the experiment must last long enough to ensure that the system has reached a steady state and data must be collected only when the system has reached a steady state. It is in most cases non-trivial to determine the proper duration of the simulation. Determining when, or whether, a model reaches steady state is a complicated issue. For instance, when simulating a highly loaded buffer, and starting the simulation with an empty buffer, the initial buffer filling level is much lower than the average filling level in the steady state. Hence, using observations made before the system has reached a steady state would bias the estimation. Therefore, when studying stationary properties the observations must start after a warm-up (transient) period. The simulation is set to terminate when a given level of confidence in the estimates has been achieved or a maximum simulation time has elapsed.

4.2.2 General principles of discrete-event simulation

This section introduces discrete-event simulation and gives a brief overview of the essential modeling concepts. For further details see [5].

Time-driven and event-driven simulations

There are two main approaches for simulating the behavior of a system described by a stochastic process.

- *Time-driven simulation.* The simulation clock is advanced in fixed increments. At every advance of the clock, the state of the system is checked and possibly modified, see Figure 4.6.
- *Event-driven simulation.* The simulation clock is advanced to the next event, see Figure 4.7.

Time-driven simulation may be efficient if the underlying process generates events at regular intervals, e.g. time-slotted systems, or if it is desirable to discretize a continuous process. However, it is less suited when the time between events may be of arbitrary length since this may result in a large processing overhead (the system state is sampled even if no event occurs; useless samples) and in missing state changes (the system state is not sampled although it changes).

Event-driven simulation skips the time when there is no change and only sample the system state when events occur, i.e. when the state of the system changes. In other

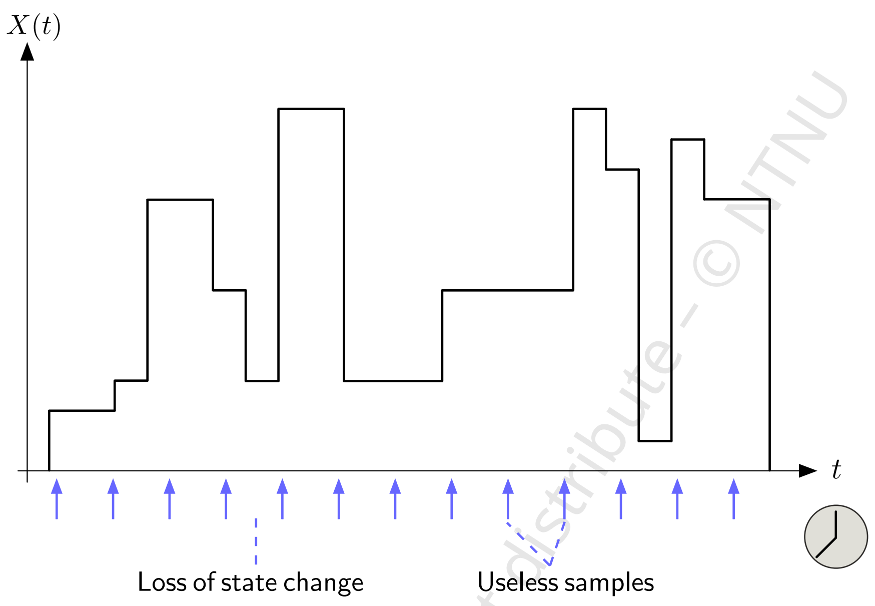


Figure 4.6: Time-driven simulation

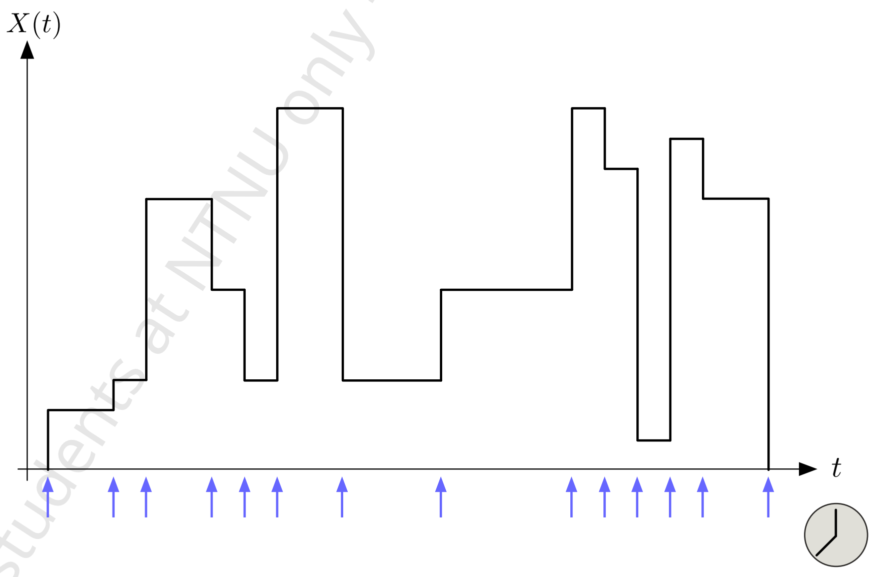


Figure 4.7: Event-driven simulation

words, the time between events does not affect the efficiency of event-driven simulations. Therefore, although it entails a more complex clock scheduling and requires handling list of events, event-driven simulation is commonly preferred as it is significantly more efficient. Note that in the case of several concurrent sub-processes, events may occur at the same point in time and care must be taken to handle them correctly.

In addition, a simpler approach can be used to simulate (semi-)Markov processes, see Section 3.3.1. For such processes each state change is a regeneration point, thus it is not necessary to keep an explicit list of events. The time to the next event may be drawn each time an event occurs. This approach is called (semi-)Markov simulation. If the steady state probabilities of the simulated Markov system can be determined, the time can be omitted completely yielding to extremely efficient simulations. This approach is referred to as K.M. Olsson [28] or Kosten [29] simulation.

Concepts in discrete-event simulation

This section defines essential concepts in discrete-event simulation [30].

- *System.* A collection of entities that interact over time.
- *Model.* An abstract representation of the system.
- *System state.* A set of variables describing the system at a time t.
- *Entity.* System component explicitly represented in the model.
- *Attribute.* Property of an entity.
- *Event.* Occurrence of change of the system state.
- *Event notice.* Record of an event to occur.
- *Event list.* List of event notices ordered by time of occurrence.
- *Activity.* Specified time duration, e.g. drawn from a probability distribution.
- *Delay.* Unspecified time duration, e.g. time in a queue.
- *Clock.* Simulated time.
- *Resource.* A passive object used by some entities to perform an activity.

These concepts constitute a framework for describing discrete-event simulation models. *Entities* have *attributes* and interact together during an *activity* which may generate *events* that result in a change of the *system state.* In addition, the adjective "endogenous" is used to describe activities and event occurring within a system and "exogenous" for activities and event in the environment that affect the system. An example is given in Figure 4.8 where the modeling concepts are used to describe a queueing system (a bank with bank clerks and a queue of customers).

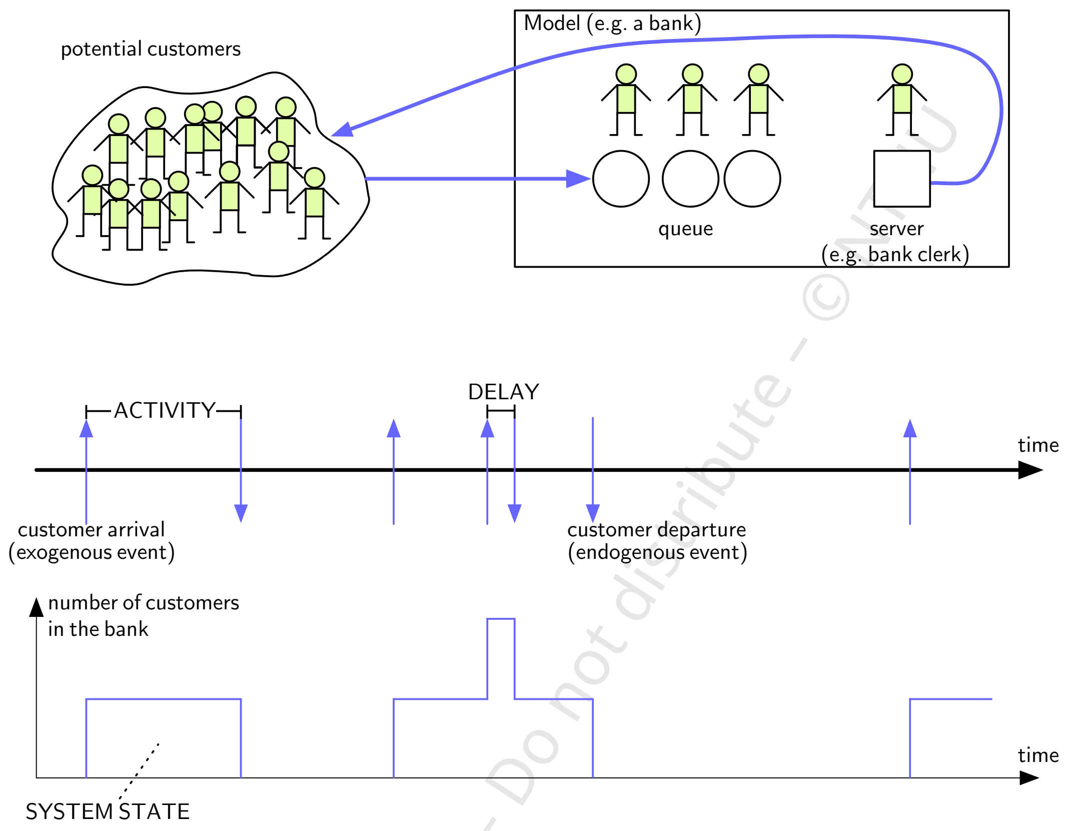


Figure 4.8: Applying the modeling concepts

Interaction between entities

Entities, or objects are key elements in a model. Entities may compete for resources, cooperate and interrupt each other. They may also generate other entities during the simulation and clone themselves (bootstrapping), see Figure 4.9.

Entities may be either *static*, i.e. created at startup time and lasting during the entire simulation, or *dynamic*, i.e. created at startup time and terminated before the simulation is over or created during the simulation. Entities perform activities which may involve waiting for one or more resources and/or for one or several other entities (rendez-vous synchronization, condition waiting).

A simulator must include control functions to handle a large number of entities, e.g. mechanisms to passivate and re-activate entities, to queue entities pending, queues, lists or tables to enable events to occur in a certain order.

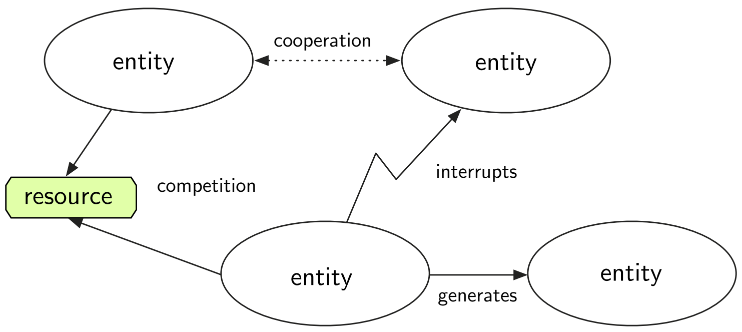


Figure 4.9: Interactions between entities

World views

Several approaches, *world views* or simulation modeling paradigms, may be used when developing a discrete-event simulation model depending on the problem to study and on the developer's experience and preferences. There are three main approaches:

- *Activity scanning* (time-driven approach). The modeling focuses on activities and the conditions for an activity to start. The advantage of this approach is its simplicity which makes it easy to implement in a general programming language. Its main drawback is its inefficiency due to the repeated scanning of all the activities.
- *Event scheduling* (event-driven approach). The modeling focuses on the events and requires a chronological list of future events to be maintained[5].
- *Process interaction.* The modeling focuses on entities and their dependences and interactions. A process describes the behavior of an entity (sequence of activities, "life cycle"), its interactions with other entities and its use of resources. It requires a chronological list of future events to be maintained and synchronization mechanisms. A system can be partitioned in subsystems and the life cycle of each subsystem can be modeled separately which makes the process oriented approach attractive when modeling large systems.

4.2.3 Simulation tools

A lot of both commercial and non-commercial simulation tools are available. The purpose of this section is not to provide an exhaustive list of all of them but rather to give a simple classification and relevant criteria for selecting a simulation tool.

[5] *List processing* is not discussed in this book. Examples include doubly-linked lists, tables, binary trees, heap trees (used in DEMOS), splay, etc.

Types of simulation tools

There are different ways of classifying simulation tools. The one used here, and also for example in [31], consists in classifying the simulation tools according to their level of details. Note that it is the abstraction level and the complexity rather than the number of lines of code that matter.

General purpose programming languages The advantage of using a general purpose programming language is that one can use a language he/she is already familiar with. In addition, there are no or few limitations on what can be done. However, the amount of time and effort required to design, code and verify a proper simulator may be overwhelming. Examples of general purpose programming languages include C, C++, Java, Python, etc.

Simulation programming languages A simulation programming language is a programming language specifically designed for simulation purposes, i.e. with built-in structures and functions for simulation. The amount of programming required is considerably reduced compared to using a general purpose programming language but is still significant. In addition, one has to learn a new programming language. Simula [32] is an example of a pure object oriented simulation programming language.

Simulation libraries A simulation library provides building blocks needed to make a simulator. These building blocks may be implemented using a general purpose programming language or simulation programming language. The programming effort is reduced to the description of the behavior of the system to simulate itself. DEMOS [24, 5], Discrete Event Modelling On Simula, is an example of simulation library. It is a context class that contains many building blocks for discrete event simulators.

Specialized simulation tools The programming effort required is further reduced when using a specialized simulation tool. In addition to the building blocks needed to make a simulator, such a tool contains models of system components that can be reused. The programming effort is then reduced to the configuration and coupling of built-in components, possibly using a graphical front-end, and writing simulation scenarios. In other words, the simulation study can be started almost immediately. The drawback is that it may often be challenging, or even impossible, to add functionalities or implement new models, in particular if the abstraction level of the model differs from the abstraction level of the tool. NS (the Network Simulator)[6] is an example of specialized simulation tool. It is a discrete event simulator targeted at networking research which provides substantial support for simulation of TCP, routing, and multicast protocols over wired

[6]http://www.nsnam.org

and wireless (local and satellite) networks. Other examples include OpNet[7], OMNet++[8], J-Sim[9], Möbius[10], etc.

Criteria for specialized simulation tool selection

Many aspects must be taken into consideration when selecting a tool for a simulation study. Some are formulated as questions hereafter.

- *Modeling.* Which modeling methods can be used? What is the level of abstraction? Is hierarchical and modular model building possible?
- *Built-in models.* Are there models that can be reused directly? For example protocols (e.g. ATM, TCP/IP, UDP, WAP), sources (e.g. phone calls, web traffic, video streaming, peer-to-peer traffic), failure processes (e.g. hardware failures, communication noise), routing protocols (e.g. OSPF, RIP) etc.
- *Simulation mechanisms.* What types of simulation are supported? How are events and the event list managed (scheduling, list processing)?
- *Setup of a simulation experiment.* Is it possible to easily set up both transient and stationary experiments? Is it easy to define start and stop criteria? Is it possible to specify the transient period? Is it possible to handle independent replicas? Is sectioning (batch-mean) supported? Is logging support provided?
- *Data collection.* How good is the support for accumulating, tracing and presenting statistics? How easy is it to tailor the data collection and presentation?
- *Adaptability.* How easy is it to add new models/functions or modify/extend the existing ones? Is it an open-source tool?
- *Efficiency.* How CPU-demanding is the execution of a simulation using the tool? Is distributed simulation supported?
- *Usability.* How good is the documentation (user guide and reference manual)? Is there any graphical input and/or output graphical front-end? Does it exist any interactive debugger?
- *Availability.* How much does it cost? Under which license is it distributed? Which support is provided? Which hardware and software platforms are supported? What are the plans for future development?
- *Type of problem.* Is the system that shall be simulated composed of standard components/protocols that shall be investigated for a (few) configurations and system loads, or is the study primarily aimed at investigating new mechanisms and/or system designs? How broad shall the study be? For instance, shall only

[7]http://www.opnet.com/
[8]http://www.omnetpp.org/
[9]http://www.j-sim.org/
[10]http://www.mobius.uiuc.edu/

packet flows in a network be studied, or does the study also include aspects like network management, fault handling, service handling and provision, logistics, etc.

- *Skill of the simulator developer/development team.* This one should be obvious! However, a warning seems appropriate. Buying an expensive tool with claimed simplicity and user friendliness does *not* substitute insight and knowledge.

5 Poisson and Markov processes

This chapter presents the *Poisson process* and its properties. In particular, it describes how Poisson processes can be aggregated to form Markov processes that are an important part of *Markov models* that will be used for performance and dependability evaluations in Chapters 6 and 7.

5.1 Poisson processes

As already mentioned in Section 1.1.2 the aggregation of a large number of processes is, under certain restrictions, a Poisson process (Palm-Khintchine's theorem). This makes Poisson processes well suited to model many physical phenomena in nature and in technical systems, e.g. ICT-systems. As it will be shown in the following, the use of Poisson processes leads to system models that have rather straightforward analytical solutions. Examples in the context of ICT-systems that are commonly modeled with Poisson processes include call arrivals to a telephone exchange, packet arrivals to an IP router, and the occurrence of failures.

5.1.1 Poisson distribution

Let the intensity λ of events in a regular point process (only one type of events and only one event at a time) be constant and independent of the previous events. Such a process is a *Poisson process*.

The probability of one event in a short interval $(t, t + \Delta t]$ is

$$P(\text{one event in } (t, t + \Delta t]) = \lambda \Delta t + o(\Delta t) \tag{5.1}$$

where $o(\Delta t)$ represents the probability of more than one event in $(t, t + \Delta t]$ and tends to 0 faster than Δt (regular point process), that is

$$\lim_{\Delta t \to 0} \frac{o(\Delta t)}{\Delta t} = 0 \tag{5.2}$$

The sign in front of $o(\Delta t)$ makes no difference because the contribution of this term is infinitely small.

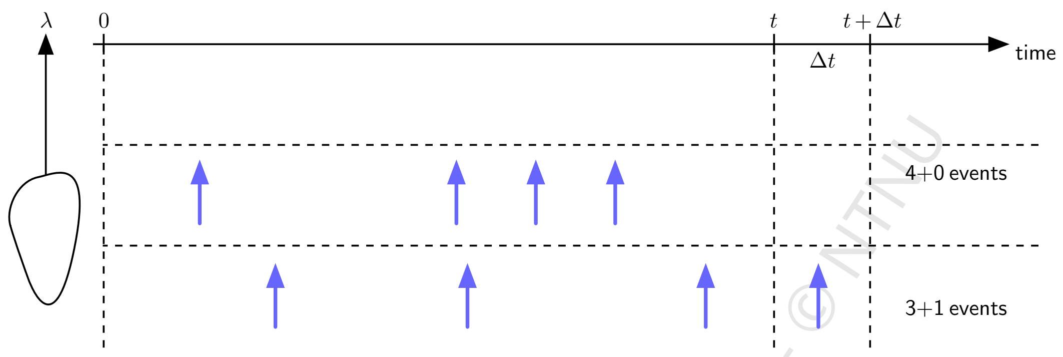


Figure 5.1: Example with 4 events in $(0,t]$ and no event in $(t,t+\Delta t]$, and 3 events in $(0,t]$ and one event in $(t,t+\Delta t]$

Using (5.1), the probability of no event in $(t,t+\Delta t]$ is

$$P(\text{no event in } (t,t+\Delta t]) = 1-\lambda\Delta t + o(\Delta t) \tag{5.3}$$

The probability of $I = i \in \mathbb{N}$ events within the time t is

$$p_i(t) = P(I = i \text{ events in } (0,t]),\ i \in \mathbb{N} \tag{5.4}$$

For $I \geqslant 1$, there are two ways of obtaining $I = i$ events in $(0,t+\Delta t]$: either

- i events have already occurred in $(0,t]$ and no new event occurs in $(t,t+\Delta t]$, or
- $i-1$ events have already occurred in $(0,t]$ and one new event occurs in $(t,t+\Delta t]$.

See Figure 5.1 for an illustration. Hence,

$$p_i(t+\Delta t) = p_i(t)(1-\lambda\Delta t+o(\Delta t)) + p_{i-1}(t)(\lambda\Delta t + o(\Delta t)),\ i \in \mathbb{N}^* \tag{5.5}$$

Since $p_i(t) \leqslant 1,\ t \geqslant 0,\ i \in \mathbb{N}$, all the product terms that include $o(\Delta t)$ can be grouped in one term, so (5.5) can be rewritten

$$p_i(t+\Delta t) = p_i(t)(1-\lambda\Delta t) + p_{i-1}(t)\lambda\Delta t + o(\Delta t),\ i \in \mathbb{N}^* \tag{5.6}$$

In the following, the $o(\Delta t)$-terms will be grouped without comments.

For $i = 0$, no event occurs in $(t,t+\Delta t]$. Eq. (5.6) applies therefore for any $i \in \mathbb{N}$, if we define $p_{-1}(t) = 0,\ \forall t$.

Rearranging and dividing (5.6) by Δt gives

$$\frac{p_i(t+\Delta t) - p_i(t)}{\Delta t} = \lambda(p_{i-1}(t) - p_i(t)) + \frac{o(\Delta t)}{\Delta t} \tag{5.7}$$

Taking the limit when Δt tends to 0, the left-hand side is equal to the time derivative of $p_i(t)$ and the last term on the right-hand side tends to 0. Thus, we obtain

$$\frac{\mathrm{d}}{\mathrm{d}t}p_i(t) = p_i'(t) = \lambda(p_{i-1}(t) - p_i(t)) \tag{5.8}$$

This is a *difference-differential equation* and its solution is the *Poisson distribution*.

$$p_i(t) = C\frac{(\lambda t)^i}{i!}e^{-\lambda t},\ t \geqslant 0,\ i \geqslant 0 \tag{5.9}$$

where C is a constant that can be determined using the normalization condition.

$$\sum_{i=0}^{\infty} p_i(t) = \sum_{i=0}^{\infty} C\frac{(\lambda t)^i}{i!}e^{-\lambda t} = 1 \implies C = 1 \tag{5.10}$$

Poisson distribution

$$p_i(t) = \frac{(\lambda t)^i}{i!}e^{-\lambda t},\ t \geqslant 0,\ i \geqslant 0 \tag{5.11}$$

The solution to (5.8) can be found by recursion, solving it first for $i = 0$. For $i = 0$, we have a first order differential equation in $p_0(t)$, $p_0'(t) = -\lambda p_0(t),\ t \geqslant 0$, that can be solved. By integrating with respect to t, we obtain:

$$\ln p_0(t) = -\lambda t + C \Leftrightarrow p_0(t) = Ce^{-\lambda t} \tag{5.12}$$

where C is a constant that can be found using that the probability of no event at time $t = 0$ is $p_0(0) = 1$. We find $C = 1$ and $p_0(t) = e^{-\lambda t}$, $t \geqslant 0$.

5.1.2 Inter-arrival times

In this section, we are interested in the distribution of the time T between two consecutive events in a Poisson process.

The probability of no event in the interval $(0, t]$ is

$$P(T > t) = \int_{x=t}^{\infty} f_T(x)\,\mathrm{d}x = p_0(t) = e^{-\lambda t} \tag{5.13}$$

Therefore, the probability density function of T is

$$f_T(t) = \frac{\mathrm{d}}{\mathrm{d}t}P(T \leq t) = \frac{\mathrm{d}}{\mathrm{d}t}(1 - P(T > t)) = -\frac{\mathrm{d}}{\mathrm{d}t}P(T > t) = \lambda e^{-\lambda t} \tag{5.14}$$

Hence, the time between two consecutive events in a Poisson process follows a negative exponential distribution.

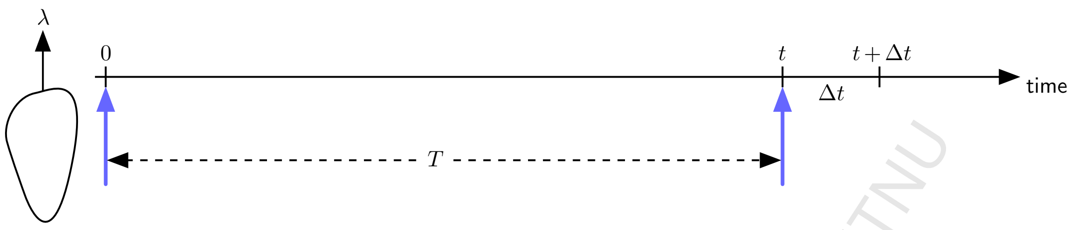


Figure 5.2: Time between two consecutive events

Inter-arrival time distribution in a Poisson process

The time between two consecutive events in a Poisson process follows a negative exponential distribution.

The inter-arrival distribution can also be obtained directly from the assumption that $P(\text{one event in } (t, t+\Delta t]) = \lambda\Delta t + o(\Delta t)$. Let the interval $(0, t]$ be subdivided in n smaller intervals of equal length $\Delta t = t/n$. The probability of no event in the interval $(0, t]$ followed by one event in $(t, t+\Delta t]$ is

$$\begin{aligned} P(t < T \le t+\Delta t) &= (1-\lambda\Delta t + o(\Delta t))^n(\lambda\Delta t + o(\Delta t)) \\ &= \left(1-\lambda\frac{t}{n}\right)^n \lambda\Delta t + o(\Delta t) \end{aligned} \tag{5.15}$$

Let $m = -n/(\lambda t) = -1/(\lambda\Delta t)$. $m \to \infty$ when $\Delta t \to 0$ and $\lim_{m\to\infty}(1+\frac{1}{m})^m = e$.

$$\begin{aligned} f_T(t) &= \lim_{\Delta t\to 0} \frac{P(t < T \le t+\Delta r)}{\Delta t}, \; t \geqslant 0 \\ &= \lim_{\Delta t\to 0}\left[\left(1+\frac{1}{m}\right)^{-m\lambda t}\lambda + \frac{o(\Delta t)}{\Delta t}\right] \\ &= \lambda e^{-\lambda t} \end{aligned} \tag{5.16}$$

In summary,

- the probability of one event in a short interval $(t, t+\Delta t]$ is: $P(\text{one event in } (t, t+\Delta t]) = \lambda\Delta t + o(\Delta t)$,
- the inter-arrival times are negatively exponentially distributed, and
- the number of events until and including time t follows a Poisson distribution.

These three properties form the *memoryless triangle* shown in Figure 5.3.

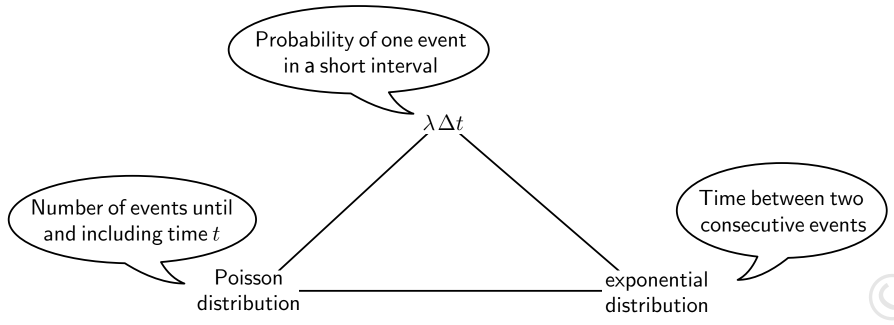

Probability of one event in a short interval
$\lambda\Delta t$
Number of events until and including time t
Time between two consecutive events
Poisson distribution
exponential distribution

Figure 5.3: The memoryless triangle

5.1.3 Memoryless property of the negative exponential distribution

We have seen in Section 5.1.2 that the time between consecutive events for a Poisson process of intensity λ follows a negative exponential distribution of parameter λ. This section addresses the memoryless property of the negative exponential distribution.

Assume that a time t_a has elapsed without any event. The time T to next event is $P(t + t_a < T \le t + t_a + \mathrm{d}t \mid T > t_a)$ and can be determined using Bayes' formula, see Section 2.1.3.

$$P(t + t_a < T \le t + t_a + \mathrm{d}t \mid T > t_a) = \frac{\lambda e^{-(t+t_a)\lambda}}{e^{-\lambda t_a}}\,\mathrm{d}t = \lambda e^{-\lambda t}\,\mathrm{d}t \tag{5.17}$$

Hence, the distribution of the time to next event is the same from time t_a as from time $t = 0$. This property is referred to as the *memoryless property* of the negative exponential distribution. This means that if you observe the time to next event from a random point of time (forward recurrence time), you always see the same distribution. This result has already been mentioned in Section 2.4.2. The memoryless property can be explained from the assumption that $P(\text{one event in } (t, t+\Delta t]) = \lambda\Delta t + o(\Delta t)$ is valid at any point of time irrespective of what has happened in the past.

The i^{th} moment of the negative exponential distribution has been determined in Example 2.7,

$$\mathrm{E}(T^i) = \int_0^\infty t^i \lambda e^{-\lambda t}\,\mathrm{d}t = \frac{i!}{\lambda^i} \tag{5.18}$$

In particular, the first and second order moments are $\mathrm{E}(T) = 1/\lambda$ and $\mathrm{E}(T^2) = 2/\lambda^2$, respectively, and thus the variance is $\mathrm{Var}(T) = \mathrm{E}(T^2) - \mathrm{E}(T)^2 = 1/\lambda^2$.

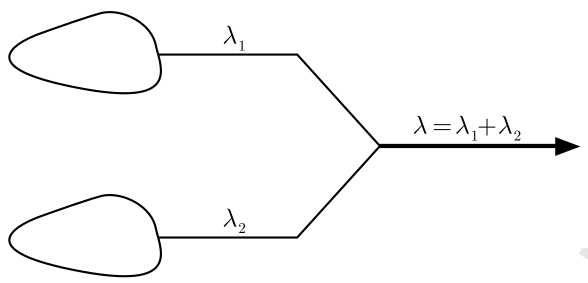


Figure 5.4: Merging two Poisson processes

5.1.4 Merging Poisson processes

Merging processes of the same type does generally not create a new process of the same type, not even if the processes are renewal processes, see Section 3.1.2. However, this is true in the case of Poisson processes; merging two or more Poisson processes results in a new Poisson process. This property greatly simplifies the analytical solution of Markov models, see Section 5.2.

This property can be explained by looking at the merging of two Poisson processes of intensity λ_1 and λ_2, respectively. The probability of no event in $(t, t+\Delta t]$ in the merged process is

$$P(\text{no event in } (t, t+\Delta t]) = (1 - \lambda_1 \Delta t + o(\Delta t))\,(1 - \lambda_2 \Delta t + o(\Delta t)) \tag{5.19}$$

Therefore, the probability of one event in $(t, t+\Delta t]$ in the merged process is

$$P(\text{one event in } (t, t+\Delta t]) = (\lambda_1 + \lambda_2)\Delta t + o(\Delta t) \tag{5.20}$$

This is exactly the probability of one event in a Poisson process of constant intensity $\lambda_1 + \lambda_2$.

The same result can be obtained by considering the probability of the number of events. Remember from the *memoryless triangle* that the number of events N generated by a Poisson process until and including time t follows a Poisson distribution.

$$P(N = n) = \frac{(\lambda t)^n}{n!} e^{-\lambda t} \tag{5.21}$$

When merging two Poisson processes, i.e. $i = 1, 2$, the number of events in the merged process is the sum of $N_1 = n_1$ events in process 1, and $N_2 = n - n_1$ events in process 2. Because both processes are Poisson processes, the events in processes 1 and 2 occur independently of each other. Hence, the distribution of the number of events in the merged process is

$$P(N_1 + N_2 = n) = \sum_{n_1=0}^{n} P(N_1 = n_1) P(N_2 = n - n_1) = \frac{((\lambda_1 + \lambda_2)t)^n}{n!} e^{-(\lambda_1+\lambda_2)t} \tag{5.22}$$

which can be recognized as the Poisson distribution with intensity $\lambda_1 + \lambda_2$. From the memoryless triangle it is then evident that the merged process is a Poisson process.

This demonstration in the case of two Poisson processes can be generalized by induction and it can be shown that this result is valid for an arbitrary number of Poisson processes

Merging of Poisson processes (Palm's theorem)

When n independent Poisson processes with intensities λ_i, $i = 1, \cdots, n$, are merged they form a Poisson process of intensity $\sum_{i=1}^{n} \lambda_i$.

If we consider a large number of sources, e.g. users of a telephony service, where each source is described by a Poisson process, then the resulting process from the collection of sources is a Poisson process. According to Palm-Khintchine's theorem (see Section 3.1.3), the aggregate process resulting from the superposition of an infinite number of processes is a Poisson process. When the individual processes are Poisson processes, this result is true for any number of processes.

5.1.5 Splitting a Poisson process

The opposite to merging Poisson processes, is splitting a Poisson process in n processes. If the splitting is random, i.e. each event from the original process results in an event in process i with a probability p_i and independently of the other events, then each of the new processes is a Poisson process.

$$P(\text{one event in process } i \text{ in } (t, t + \Delta t]) = \lambda p_i \Delta t + o(\Delta t) \tag{5.23}$$

Splitting of Poisson processes

When a Poisson process of intensity λ is randomly split in n processes, these processes are Poisson processes with intensities $p_i\lambda$, $i = 1, \cdots n$, respectively, where p_i is the probability of selecting process i.

Note that this result is only valid if the splitting is random. Consider, for instance, the distribution of time between events in process i if the events are split in a round-robin fashion, i.e. the first event in the original process results in an event in process 1, the second event causes an event in process 2, etc. The time between events in process i is then Erlang-n distributed if the original process is split in n processes.

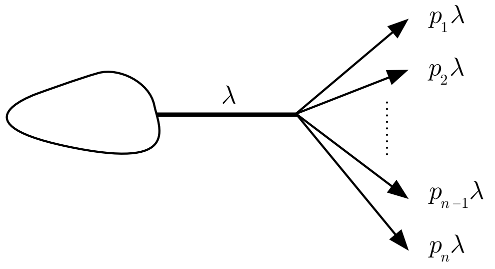


Figure 5.5: Splitting a Poisson process

5.1.6 Probability that a given process generates an event first

Consider two Poisson processes of intensity λ_1 and λ_2, respectively. We are interested in the probability that the next event in the merged process is from process 1. Let T_1 and T_2 denote the time to the next event in processes 1 and 2, respectively. p_1 is the probability that the next event in process 1 occurs in the interval $(t, t + \mathrm{d}t]$ and that no event in the process 2 occurs before t. Hence,

$$\begin{aligned} p_1 &= P(\text{next event is from process 1}) \\ &= \int_{t=0}^{\infty} P(t < T_1 \leqslant t + \mathrm{d}t \cap T_2 > t) \\ &= \int_{t=0}^{\infty} P(t < T_1 \leqslant t + \mathrm{d}t) P(T_2 > t) \\ &= \int_{t=0}^{\infty} \lambda_1 e^{-\lambda_1 t} e^{-\lambda_2 t}\,\mathrm{d}t = \frac{\lambda_1}{\lambda_1 + \lambda_2} \end{aligned} \tag{5.24}$$

The probability p_1 that the next event is from process 1 is the intensity of process 1 divided by the intensity of the merged process. This result can be generalized to an arbitrary number of Poisson processes.

Next event from parallel Poisson processes

Let λ_i , $i = 1, \cdots , n$ be the intensity of process i in a merged Poisson process consisting of n processes. The probability p_i that the next event is from process i is

$$p_i = P(\text{next event is from process } i) = \frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j} \tag{5.25}$$

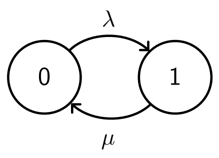


Figure 5.6: State transition diagram of a two-state Markov model

5.2 Markov processes

A Markov process, as defined in Section 3.3.1, can be regarded as a process where the events are generated by a collection of Poisson processes that may be either active or passive. The set of *active processes* is given by the state of the system. In this book, we focus on continuous time discrete space processes, i.e. the events that are considered may occur along a continuous time axis and will change the system state in discrete steps.

Analytical models are rather simple when all the processes are Poisson processes (*Markov models*):

- the time to the next event is negatively exponentially distributed with parameter equal to the sum of the intensities of the active processes, see Section 5.1.2,
- the succeeding sequence of events does not depend on the time the system has spent in the current state (memoryless property of the negative exponential distribution), see Section 5.1.3, and
- the next event can easily be determined when the *active processes* are known, see Section 5.1.6.

5.2.1 Two-state Markov process

The simplest Markov model consists of only two states denoted 0 and 1, respectively. Consider for instance the PC system studied in Example 3.5 which is either working (state 0) or failed (state 1). The system state changes from state 0 to 1 (e.g. a failure has occurred) with an intensity λ and from state 1 to 0 (e.g. the system is repaired) with intensity μ. The processes in the system are Poissonian which means that the system dynamics are governed by a Markov process. In state 0 the active Poisson process has intensity λ; in state 1 the active Poisson process has intensity μ. An illustration of this two-state Markov model is shown in Figure 5.6.

This simple example is used in the remaining of this section to detail how to find the probability of observing the system in a given state at time t.

Let $I(t)$ be the state of the system at time t.

$$I(t) = \begin{cases} 1 \text{ if the system is in state 1 at time } t \\ 0 \text{ if the system is in state 0 at time } t \end{cases} \tag{5.26}$$

As mentioned above, the system changes from state 0 to 1 according to a Poisson process of intensity λ and from state 1 to 0 according to another Poisson process of intensity μ.

Let $p_i(t)$ be the probability that the system is in state i ($i = 0, 1$) at time t, given that the system is in state 0 at time $t = 0$.

$$p_i(t) = P(I(t) = i \mid I(0) = 0) = P(I(t) = i) \tag{5.27}$$

The probability $p_0(t + \Delta t)$ that the system is in state 0 at time $t + \Delta t$ is equal to the probability that the system was in state 0 at time t and no new events occurred in $(t, t + \Delta t]$ or that the system was in state 1 and one event occurred in $(t, t + \Delta t]$.

$$p_0(t + \Delta t) = p_0(t)(1 - \lambda \Delta t) + p_1(t)\mu \Delta t + o(\Delta t) \tag{5.28}$$

Taking the limit when Δt tends to 0, we obtain

$$p_0'(t) = -\lambda p_0(t) + \mu p_1(t) \tag{5.29}$$

Using the normalization condition, we substitute $p_1(t)$ by $1 - p_0(t)$ in (5.29)

$$p_0'(t) = -\lambda p_0(t) + \mu(1 - p_0(t)) \tag{5.30}$$

This is a first order linear differential equation which solution is

$$p_0(t) = Ce^{-(\lambda+\mu)t} + \frac{\mu}{\lambda + \mu} \tag{5.31}$$

where C denotes a constant that can be found using the initial condition i.e. the system is in state 0 at time $t = 0$, $p_0(0) = 1$.

$$p_0(0) = Ce^{-(\lambda+\mu)0} + \frac{\mu}{\lambda + \mu} = C + \frac{\mu}{\lambda + \mu} = 1 \Leftrightarrow C = \frac{\lambda}{\lambda + \mu} \tag{5.32}$$

Transient solution

$$p_0(t) = \frac{\lambda}{\lambda + \mu} e^{-(\lambda+\mu)t} + \frac{\mu}{\lambda + \mu} \tag{5.33}$$

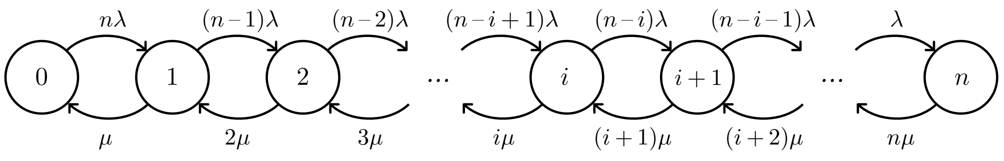


Figure 5.7: State transition diagram of a model with n sources.

$p_0(t)$ decreases exponentially with time, and the time constant is $\frac{1}{\lambda+\mu}$. When the time t tends to infinity, the probability of being in state 0 is independent of the initial state at time $t = 0$. The transient term vanishes and we have the *stationary (steady state) solution*.

Stationary solution

$$p_0(\infty) = \frac{\mu}{\lambda+\mu} = \frac{1/\lambda}{1/\lambda + 1/\mu} \tag{5.34}$$

$1/\lambda$ and $1/\mu$ are the expected time in state 0 and 1, respectively. Hence, the stationary probability of state 0 is equal to the expected time in state 0 divided by the expected *cycle time;* i.e. the sum of expected time in state 0 and 1.

Example 5.1 Model of speech sources
The two-state model is for instance used to model speech sources. Each source is considered to be either active (state 1) or silent (state 0). In state 1 a source is generating a bit stream (e.g. 64 kbit/s) and in state 0 no bit is generated. The duration of each state is n.e.d. and each state change is governed by a Poisson process. If there are n speech sources, it is sufficient to keep track of the number of active (or silent) sources. When i sources are active and $n-i$ sources are silent, there are two active Poisson processes: a (composite) process of intensity $i\mu$ and a (composite) process of intensity $(n-i)\lambda$. The system state can be defined as the number of active (or silent) sources. If all the sources are independent of each other, the stationary probability of having i active sources at an arbitrary point of time is given by the binomial distribution (see Table 2.1 on page 71) with $p = p_1(\infty) = \lambda/(\lambda+\mu)$.

5.2.2 State transition diagrams (Markov chains)

It is convenient to express the states of a Markov process by state transition diagrams as shown in Figure 5.6 and 5.7.

A state transition diagram, or *Markov chain*, is a directed graph where the nodes represent the states and the directed arcs the possible transitions between the states. A node may contain additional details about the system state. It is common to assign a unique number to each state to simplify state reference when establishing the system of state equations. The state transition diagram in Figure 5.7 illustrates the model described in Example 5.1. In this case the system state is the number of active users in the system.

The directed arcs are labeled with the intensity of the Poisson processes that enforce the transition. For instance, in state i in Figure 5.7, the intensity $i\mu$ on the transition from state i to state $i-1$ refers to the total intensity of the i active Poisson processes that may complete their session and become inactive, i.e. enforce a reduction in the number of active sources. Likewise, the intensity $(n-i)\lambda$ on the transition from state i to state $i+1$ refers to the total intensity of the Poisson processes that may start a new session and hence increase the number of active sources.

5.2.3 General Markov model

In the general case, there is an arbitrary number of outgoing transitions from a state i in a Markov model. Let q_{ij} denote the intensity of a transition from state i to state j. The total intensity out of state i, i.e. the intensity of the (composite) Poisson process that enforces the system to leave state i, is $\sum_{\forall j, j\neq i} q_{ij}$.

State sojourn time distribution

In a Markov model, the sojourn time in state i follows a negative exponential distribution with parameter $\sum_{\forall j, j\neq i} q_{ij}$.

$$S_i \sim \text{n.e.d.}\left(\sum_{\forall j, j\neq i} q_{ij}\right) \tag{5.35}$$

Transition probability

In a Markov model, the probability of transition from state i to state j given that a transition occurs is equal to the probability that the first event occurs in the Poisson process of intensity q_{ij} that enforces this transition.

$$p_{ij} = P(\text{next transition from state } i \text{ is to state } j) = \frac{q_{ij}}{\sum_{\forall j, j\neq i} q_{ij}},\ j \neq i \tag{5.36}$$

5.2.4 Stationary solution

This section describes how the stationary solution can be determined without first finding the transient solution as described previously.

In this book, systems of linear state equations are used, however this section also shows briefly how the matrix form solution can be applied to finding the stationary state probabilities.

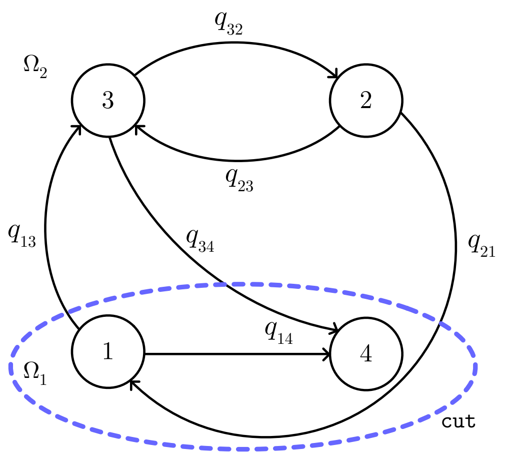


Figure 5.8: General state transition diagram

System of linear state equations

Let p_i be the stationary probability of finding the system in state i at an arbitrary point of time.

$$p_i = \lim_{t\to\infty} P(I(t) = i),\ i = 1, \ldots, n \tag{5.37}$$

where $I(t)$ denotes the system state at time t and n the number of states (n may be infinite).

To establish the system of state equations, the state space Ω is partitioned into two disjoint subspaces Ω_1 and Ω_2, i.e. $\Omega_1 \cap \Omega_2 = \emptyset$ and $\Omega_1 \cup \Omega_2 = \Omega$. An example with 4 states is given in Figure 5.8. The state space is $\Omega = \{1, 2, 3, 4\}$ and the `cut` illustrated in the figure partitions the state space in $\Omega_1 = \{1, 4\}$ and $\Omega_2 = \{2, 3\}$.

If a system in a steady state is observed over a long period of time $(0, \tau)$ where τ is large, the fraction of time the system is in state i is equal to $p_i\tau$. The expected number of transitions from state i to state j in the same period is $q_{ij}p_i\tau$ because the state transitions are governed by Poisson processes.

The expected number of transitions from Ω_1 to Ω_2 is

$$\sum_{i\in\Omega_1}\sum_{j\in\Omega_2} q_{ij}p_i\tau \tag{5.38}$$

and likewise from Ω_2 to Ω_1

$$\sum_{i\in\Omega_1}\sum_{j\in\Omega_2} q_{ji}p_j\tau \tag{5.39}$$

Over a long time period $(0, \tau)$ the system is in equilibrium, i.e. the number of transitions from Ω_1 to Ω_2 is equal to (or differs at most by 1 from) the number of transitions from Ω_2 to Ω_1.

Equilibrium equation

$$\sum_{i\in\Omega_1}\sum_{j\in\Omega_2} q_{ij}p_i = \sum_{i\in\Omega_1}\sum_{j\in\Omega_2} q_{ji}p_j \tag{5.40}$$

To establish the system of state equations required to determine the state probabilities $\pi = \{p_1, \cdots, p_n\}$, the state space is partitioned in $\Omega_1 = \{i\}$ and $\Omega_2 = \{j\}_{j=1,j\neq i}^{n}$ and the equilibrium equation is applied systematically for all $i \in \Omega$. The system of n equations and n unknowns is however still an under-defined problem because one equation can be eliminated by linear combination. To be able to solve the system of equations, we use the normalization condition $\sum_{i\in\Omega} p_i = 1$ as the last equation.

Matrix form

As an alternative to the system of linear equations given above, a matrix form can be applied.

Let $\underline{Q}$ be the transition intensity matrix.

$$\underline{Q} = \{q_{ij};\ i = 1, \cdots, n;\ j = 1, \cdots, n\} \tag{5.41}$$

where $q_{ii} = -\sum_{j=1,j\neq i}^{n} q_{ij}$.

The stationary state probability vector is

$$\underline{\pi} = \{p_1, \cdots, p_n\} \tag{5.42}$$

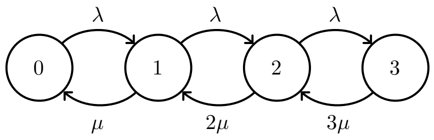
Using the same systematic partitioning of the state space as explained above, the set of equations becomes

$$\underline{0} = \underline{\pi}\underline{Q} \tag{5.43}$$

where $\underline{0}$ denotes the null vector.

The determinant of $\underline{Q}$ is 0, which means that no unique solution exists. Hence, as previously, the normalization condition $\sum_{i\in\Omega} p_i = 1$ must be used.

Example 5.2 We want to find stationary state probabilities for the following Markov model.



The transition intensity matrix $\underline{Q}$, null vector $\underline{0}$, and state vector $\underline{\pi}$ are

$$\underline{Q} = \begin{bmatrix} -\lambda & \lambda & 0 & 0 \\ \mu & -(\mu+\lambda) & \lambda & 0 \\ 0 & 2\mu & -(2\mu+\lambda) & \lambda \\ 0 & 0 & 3\mu & -3\mu \end{bmatrix}; \underline{0} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}; \underline{\pi} = \{p_0, p_1, p_2, p_3\}$$

The determinant of $\underline{Q}$ is equal to zero and $\underline{0} = \underline{\pi}\underline{Q}$ does not have a unique solution. The normalization condition is introduced by substituting the last column $\underline{Q}$ with the unity vector and changing the corresponding element of $\underline{0}$.

$$\underline{Q}_n = \begin{bmatrix} -\lambda & \lambda & 0 & 1 \\ \mu & -(\mu+\lambda) & \lambda & 1 \\ 0 & 2\mu & -(2\mu+\lambda) & 1 \\ 0 & 0 & 3\mu & 1 \end{bmatrix}; \underline{0}_n = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}; \underline{\pi} = \{p_0, p_1, p_2, p_3\}$$

Now, $\underline{\pi}$ can be determined by solving $\underline{0}_n = \underline{\pi}\underline{Q}_n$.

5.2.5 Recurrence time

The *recurrence time* R_i of a state i is the time from entering this state until the first return[1]. Imagine a "token" that circulates in the model and represents the current state of the system. At an instant of time t a state i has either 0 or 1 token and the expected number of tokens in state i is equal to the steady state probability p_i that the system is in state i. The expected recurrence time $\mathrm{E}(R_i)$ is obtained by applying Little's formula:

$$\mathrm{E}(\text{number of tokens in state } i) = p_i = \frac{1}{\mathrm{E}(R_i)}\mathrm{E}(S_i) \tag{5.44}$$

where $\mathrm{E}(S_i) = \dfrac{1}{\sum_{\forall j, j\neq i} q_{ij}}$ is the expected sojourn time in state i.

Expected recurrence time

$$\mathrm{E}(R_i) = \frac{\mathrm{E}(S_i)}{p_i} = \frac{1}{p_i \sum_{\forall j, j\neq i} q_{ij}} \tag{5.45}$$

Example 5.3 Recurrence time for the two-state Markov model
Consider the two-state Markov model illustrated in Figure 5.6. We have $q_{01} = \lambda$ and, from (5.34), when the system is stationary,

$$p_0 = \frac{\mu}{\lambda+\mu}$$

Hence,

$$\mathrm{E}(R_0) = \frac{1}{\lambda\frac{\mu}{\lambda+\mu}} = \frac{1}{\lambda} + \frac{1}{\mu}$$

Note that, in this simple case, this result can be seen directly since the recurrence time for both states is equal to the sum of the expected sojourn times in each state.

[1]The recurrence time may also be defined as the time between subsequent departures.

6 Traffic models

Traffic models represent service systems. A service system is a system which has a pool of *servers* and may have a *queue*. It receives and either accepts or rejects *service requests* generated by *sources*. A service request in the system is called a *customer*. A source is said to be *busy* when a customer from this source is in the system, otherwise it is *idle*. The system needs to allocate a number of servers to serve a customer. In this chapter, one customer is served by one server. An allocated server is released when the service is completed. A server serving a customer is said to be *busy*, otherwise it is *idle*. If no server is idle for a service request when it is received, the customer is queued, provided that a *queue* exists and has vacant queueing positions, otherwise the service request is *lost* (*blocked*).

An example of such system is a PSTN (Public Switched Telephony Network) telephone exchange which allocates telephone lines (servers) to telephone calls (customers). Call attempts (service requests) are generated by subscribers (sources). If no line is available the call attempt is lost. When a call terminates the telephone line is released.

In this chapter a few, simple, but commonly used, birth-death traffic models are introduced and the performance of the systems is analyzed, e.g. the carried traffic or the time congestion. Section 6.1 introduces queueing models and traffic characteristics and defines the notations used in this chapter. Common cases are presented in Sections 6.2, 6.3, 6.4, 6.5 and 6.6 to illustrate these concepts and to show how to analyze the models and determine the traffic characteristics.

6.1 Concepts and notations

6.1.1 Queueing model

In this book all the traffic models are referred to as *queueing models*, even if the queueing capacity is null (no queue). The reason for this is that it is convenient to use the same notation and modeling approach in systems with and without a queue. The queueing model consists of n servers and $q_{\max} = k - n \geqslant 0$ queueing positions, where k is the maximum number of customers in the system. If $k = n$ the system is a blocking system with no queue. Service requests arriving to the system when all servers are busy enter the queue and wait for a server to become idle; if the queue is full, or if there is no queue, the service requests are lost.

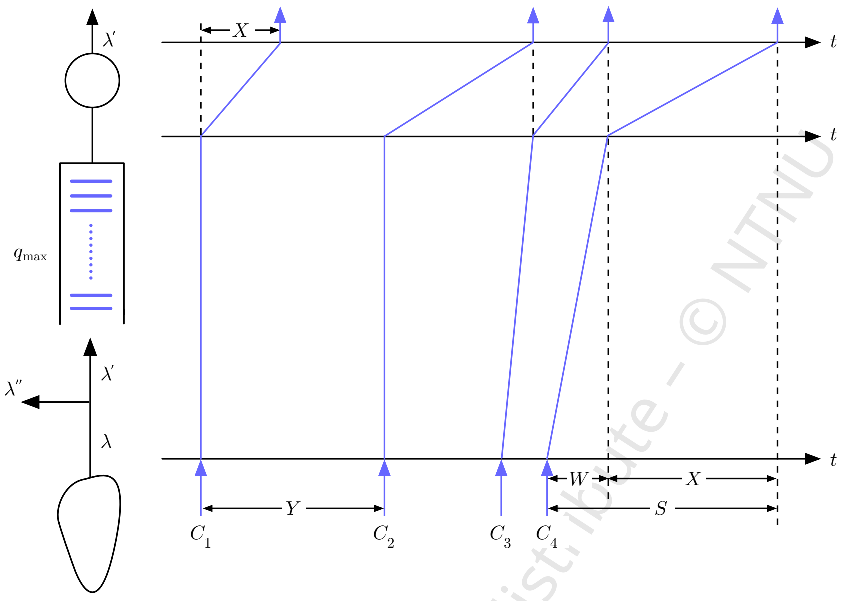


Figure 6.1: Time diagram for a single server queueing model

Let $I(t)$ be the number of customers in the system, i.e. the number of busy servers plus the number of occupied queueing positions, if any, at time t. $I(t)$ defines the state of the system and $p_i = P(I(t) = i)$, $i = 0, \ldots, k$, $\forall t$ is the steady state probability (see Section 3.2.1) that the system is in state i. In the case of *birth-death processes*, the state of the system only changes by ± 1 every time an event occurs. The only possible state transitions are from i to $i+1$ (arrival) or from i to $i-1$ (service completion). The arrival intensity in state i is denoted λ_i.

In Figure 6.1 the time instances for arrivals, queueing, and service are illustrated for a set of customers, C_i. The use of such time sequence diagrams is convenient to define the model variables.

λ	offered arrival intensity
λ'	carried arrival intensity
λ''	loss intensity
Y	inter-arrival time
S	sojourn time in system (total time in the system)
W	waiting/queueing time (time in queue),
X	service time

As mentioned, service requests are lost when arriving while all servers are busy and no queueing position is vacant. Hence, $\lambda' = \lambda - \lambda''$. All customers entering the system will eventually be served and hence the departure intensity is also λ'. Little's formula, see Section 3.4, provides a relation between the average number of customers in the system and the average time in the system.

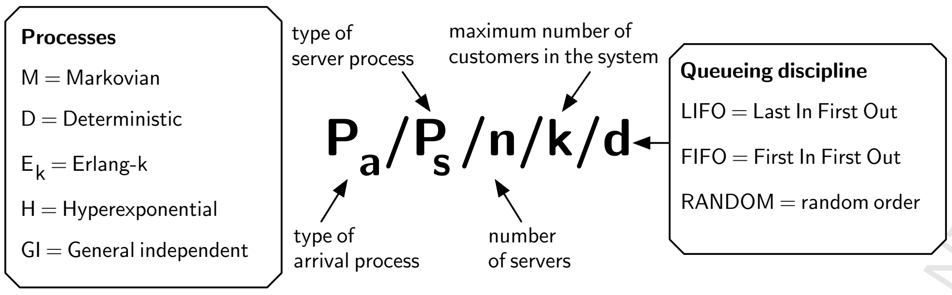


Figure 6.2: Kendall's notation

6.1.2 Kendall's notation

Kendall's notation is a common and compact representation of queueing models. As pointed out in the previous section, a model does not need to have a queue to be classified as a queueing model.

Kendall's notation

Kendall's notation describes a queueing model using five parameters written as:

$$P_a/P_s/n/k/d \tag{6.1}$$

where P_a denotes the arrival process, P_s the server process, n the number of servers, k the maximum number of customers in the system, and d the queueing discipline.

Figure 6.2 gives typical types of processes and queueing disciplines used for modeling of ICT systems. When the maximum number of customers in the system is infinite, it is commonly omitted in the representation.

In this chapter, only M/M-systems are presented, i.e. systems with Poisson arrival processes and negative exponential service times. A system with Poisson arrival processes, negative exponential service times and an infinite number of servers is represented using Kendall's notation as $M/M/\infty$, a system with a finite number n of severs and no queue (loss system) as $M/M/n/n$, and a system with n servers and an infinite number of queueing positions (delay system) as $M/M/n$. Theses systems are treated in details in the following sections of this chapter. Other examples found in the literature but beyond the scope of this book include $M/G/1/\infty$, for a system with Poisson arrivals, general service process, a single server and an infinite number of queueing positions, and $GI/M/n/\infty$, for a system with independent but general arrival process, negative exponential service times, n servers and an infinite number of queueing positions.

Example 6.1 Slotted system
Each time-slot in a time division multiplexed (TDM) system can be considered as a server. A TDM system with n time-slots is then a system with n servers. If we assume that conversations

(call sessions) are transferred over a TDM system then a time-slot is allocated per conversation. If we can assume a large number of users, the arrival of conversations follows a Poisson process (Palm-Khintchine's theorem, see Section 3.1.3). If in addition the conversation times are negatively exponentially distributed then the TDM system can be modeled as an $M/M/n/n$-system.

Example 6.2 Transmission channel
Assume that packets are transmitted on a transmission channel with capacity c [bits/s]. The packet size is n.e.d. with average size B [bits]. The packet transmission time is $X = B/c$ [sec]. If the packet arrival process is a Poisson process and the transmission buffer is large (no packets are lost), the system can be modeled as an $M/M/1/\infty$-system.

6.1.3 Two types of Markovian traffic

In this chapter, we regard the case when, i.e. we make the assumption that, each source generates service requests according to a process with constant intensity γ when it is idle (no customer from this source is in the system) and 0 when it is busy (one customer from this source is in the system).

We then distinguish between two types of traffics:

- *Erlang case, Pure Chance Traffic type One (PCT-I) or random traffic.* In this case, the number s of sources is assumed to be very large and the intensity per source is assumed to be very small. Formally, we regard the limit when $s \to \infty$ and $\gamma \to 0$. The resulting aggregate arrival process is then a Poisson process with intensity $\lambda = \lim_{s\to\infty,\,\gamma\to 0} s\gamma$. For any state i of the system (i customers in the system), the total arrival intensity is $\lambda_i = \lim_{s\to\infty,\,\gamma\to 0}(s-i)\gamma = \lim_{s\to\infty,\,\gamma\to 0} s\gamma = \lambda$. In other words, the total arrival intensity is *independent* of the state of the system. Note that, since the number of sources is assumed to be *infinite*, according to the Palm-Khintchine's theorem, see Section 3.1.3, the arrival process will also be a Poisson process when individual source inter-arrival times follow arbitrary renewal distributions.
- *Engset case, Pure Chance Traffic type Two (PCT-II) or pseudo-random traffic.* The number s of sources is assumed to be *finite*. In this case the arrival intensity depends on the state of the system. If i customers are in the system (i sources are *busy*), only $s-i$ sources can generate new service requests ($s-i$ *idle* sources), i.e. $\lambda_i = (s-i)\gamma$. The global arrival process is not in this case a renewal process, and a fortiori not a Poisson process.

From the above, we see that the Erlang case is a special case of the Engset case.

6.1.4 Traffic characteristics

To evaluate the performance of a system, several characteristics must be considered. We recall here relevant traffic characteristics introduced in Chapter 1.

4

- *Offered traffic, A:* mean number of busy servers if the number of servers in the system were infinite.
- *Carried traffic, A':* mean number of busy servers.
- *Lost traffic, A'':* difference between the offered traffic and the carried traffic.
- *Time congestion, E:* fraction of time all the resources are busy (i.e. the number of customers in the system is equal to the maximum number of customers possible in the system).
- *Call congestion, B:* probability that a service request is lost (blocked) because all the resources are busy.

Erlang (dimensionless) is the unit for traffic and Erlang-hour for traffic volumes.

Example 6.3
Let $I(t)$ be the server utilization observed over the interval (t_1, t_2). The carried traffic A' is given by

$$A' = \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} I(t)\,\mathrm{d}t \text{ [Erlang]}$$

and the carried traffic volume over the interval (t_1, t_2) by

$$\int_{t_1}^{t_2} I(t)\,\mathrm{d}t \text{ [Erlang} - \text{hour]}$$

6.1.5 Determination of traffic characteristics in analytical models

By definition, the offered traffic A is the expected number of busy servers if the number of servers were infinite. Using Little's formula, it is therefore the product of the mean arrival intensity, $\sum_{i=0}^{\infty} \lambda_i \, p_i|_{n=\infty}$ and the expected service time, $\mathrm{E}(X)$, where $p_i|_{n=\infty}$ denotes the steady state probability that the system is in state i if the number of servers were infinite.

Offered traffic

$$A = \sum_{i=0}^{\infty} \lambda_i \, p_i|_{n=\infty} \, \mathrm{E}(X) \tag{6.2}$$

In the case of PCT-I traffic (Erlang case), $\lambda_i = \lambda$, $\forall i$. Hence,

Offered traffic in the Erlang case

$$A = \sum_{i=0}^{\infty} \lambda \, p_i|_{n=\infty} \, \mathrm{E}(X) = \lambda \mathrm{E}(X) \tag{6.3}$$

Let now N be the number of service requests generated by a Poisson process of intensity λ during a service period X. The mean number of service requests generated during a service period can be calculated using the law of total expectation (2.57):

$$\mathrm{E}(N) = \int_0^\infty \mathrm{E}(N \mid X = x) f_X(x)\,\mathrm{d}x = \int_0^\infty \lambda x f_x(x)\,\mathrm{d}x = \lambda \mathrm{E}(X) \tag{6.4}$$

Hence, in the Erlang case, the offered traffic is equal to the mean number of service requests generated during a service period. Note that this result is valid without any assumption on the distribution of the service times.

When the number s of sources is assumed finite (Engset case), it is convenient to use the *offered traffic per source* and the *offered traffic per idle source*.

The offered traffic per source is the steady state probability that a source is busy when the number of servers n is equal to the number of sources, $n = s$.

Offered traffic per source

$$\alpha = \frac{\mathrm{E}(X)}{\mathrm{E}(X) + 1/\gamma} = \frac{\gamma}{\gamma + 1/\mathrm{E}(X)} \tag{6.5}$$

The offered traffic per idle source is the offered traffic per source if a source could generate service requests at any time, i.e. was always idle.

Offered traffic per idle source

$$\beta = \gamma \mathrm{E}(X) = \frac{\alpha}{1-\alpha} \tag{6.6}$$

The (total) offered traffic in the Engset case is the product of the offered traffic per source and the number of sources.

Offered traffic in the Engset case

$$A = s\alpha = \frac{s\beta}{\beta + 1} \tag{6.7}$$

By definition the carried traffic is the expected number of busy servers. Thus, it is calculated as

Carried traffic

$$A' = \sum_{i=0}^{k} \min(i, n) \cdot p_i \tag{6.8}$$

where k denotes the maximum number of customers in the system and n the number of servers.

If $n \to \infty$ then $A' \to A$ and $A'' \to 0$.

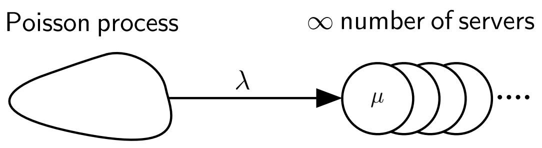


Figure 6.3: An $M/M/\infty$ queueing model

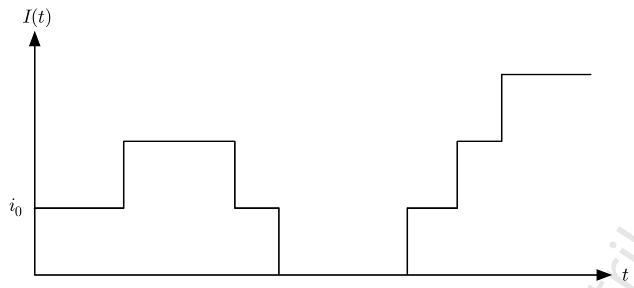


Figure 6.4: Sample path for $I(t)$

6.2 Infinite server systems (M/M/∞)

In this section, we consider systems with Poisson arrivals with intensity λ (PCT-I traffic), negative exponential service times with rate μ and an infinite number of servers ($M/M/\infty$ model). This model is a good approximation when the system consists of a large number of servers and is illustrated in Figure 6.3.

The number of customers in the system is equal to the number of busy servers because the number of servers is infinite. The probability that i servers are busy at time t is

$$p_i(t) = P(I(t) = i \mid I(0) = i_0),\ i \in \mathbb{N} \tag{6.9}$$

where i_0 is the number of busy servers at time $t = 0$. A sample path for $I(t)$ is given in Figure 6.4. Remember that the state of the system only change by $+1$ or -1 every time an event occurs.

The state transition diagram (see Section 5.2.2) for the $M/M/\infty$ model is given in Figure 6.5.

6.2.1 Transient solution

A service in progress is completed in an small interval $(t, t + \Delta t]$ with the probability

$$P(\text{service completion in } (t, t + \Delta t]) = \mu\Delta t + o(\Delta t) \tag{6.10}$$

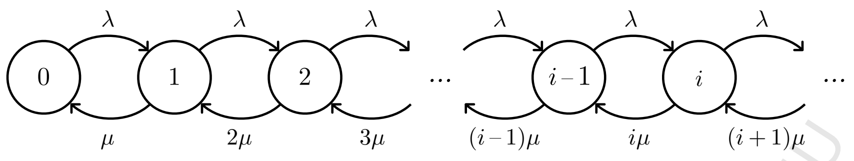


Figure 6.5: State transition diagram for the $M/M/\infty$ model of Figure 6.3

where $o(\Delta t)$ represents the probability of more than one event in $(t, t + \Delta t]$ and tends to 0 faster than Δt (regular point process).

Merging i Poisson processes results in a Poisson process with an intensity equal to the sum of the intensities of the i processes, see Section 5.1.4. So for i simultaneous services in progress, each with intensity μ, the probability of one service completion in $(t, t + \Delta t]$ (transition from i to $i - 1$) is

$$P(\text{one service completion in } (t, t + \Delta t] \mid I(t) = i) = i\mu\Delta t + o(\Delta t),\ \forall i \in \mathbb{N}^* \qquad (6.11)$$

In other words, the service completion rate in state i is $i\mu$.

The arrival intensity λ is independent of the state as shown in Figure 6.5. Hence the probability of one arrival in $(t, t + \Delta t]$ (transition from i to $i + 1$) is

$$P(\text{one arrival in } (t, t + \Delta t] \mid I(t) = i) = \lambda\Delta t + o(\Delta t),\ \forall i \in \mathbb{N}^* \qquad (6.12)$$

The probability of no events in $(t, t + \Delta t)$ is therefore

$$P(\text{no events in } (t, t + \Delta t] \mid I(t) = i) = 1 - \lambda\Delta t - i\mu\Delta t + o(\Delta t),\ \forall i \in \mathbb{N}^* \qquad (6.13)$$

The probability that the system is in state $i > 0$ at $t + \Delta t$ is given by the probabilities that the system is in either of the following states at time t and the corresponding event occurred:

$I(t) = i$	and no events in $(t, t + \Delta t]$
$I(t) = i - 1$	and one arrival in $(t, t + \Delta t]$
$I(t) = i + 1$	and one service completion in $(t, t + \Delta t]$

That is

$$p_i(t+\Delta t) = p_i(t)(1-(\lambda+i\mu)\Delta t)+p_{i+1}(t)(i+1)\mu\Delta t+p_{i-1}(t)\lambda\Delta t+o(\Delta t),\ \forall i \in \mathbb{N}^* \quad (6.14)$$

Rearranging and dividing by Δt

$$\frac{p_i(t + \Delta t) - p_i(t)}{\Delta t} = -p_i(t)(\lambda + i\mu) + (i+1)\mu p_{i+1}(t) + \lambda p_{i-1}(t) + \frac{o(\Delta t)}{\Delta t},\ \forall i \in \mathbb{N}^* \quad (6.15)$$

Taking the limit when $\Delta t \to 0$, we obtain

$$p'_i(t) = -p_i(t)(\lambda + i\mu) + (i+1)\mu p_{i+1}(t) + \lambda p_{i-1}(t),\ \forall i \in \mathbb{N}^* \tag{6.16}$$

The state $i = 0$ must be handled as a special case because the system cannot be in state -1 at time t.

$$p_0(t + \Delta t) = p_0(t)(1 - \lambda\Delta t) + p_1(t)\mu\Delta t + o(\Delta t) \tag{6.17}$$

and similarly we obtain

$$p'_0(t) = -p_0(t)\lambda + \mu p_1(t) \tag{6.18}$$

Note that by defining $p_{-1}(t) = 0$, (6.18) can be obtained directly from (6.16).

6.2.2 Stationary solution

When the system is stationary the state probabilities are independent of time or, to put it another way, the probability distribution $p_i = \lim_{t\to\infty} p_i(t)$ is independent of the initial state of the system. Thus

$$p'_i(t) = 0,\ \forall i \in \mathbb{N} \tag{6.19}$$

Equation (6.16) becomes

$$p_{i+1} = \frac{(\lambda + i\mu)p_i - \lambda p_{i-1}}{(i+1)\mu},\ \forall i \in \mathbb{N}^* \tag{6.20}$$

and from (6.18) we have

$$p_1 = \frac{\lambda}{\mu} p_0 \tag{6.21}$$

Solving (6.20) we find

$$p_i = p_0 \frac{(\lambda/\mu)^i}{i!} = p_0 \frac{A^i}{i!},\ \forall i \in \mathbb{N} \tag{6.22}$$

where $A = \lambda/\mu$ is the offered traffic, see (6.3) ($\mathrm{E}(X) = 1/\mu$).

The normalization condition $\sum_{i=0}^{\infty} p_i = 1$ gives $p_0 = e^{-A}$. Therefore the solution is the Poisson distribution with parameter A, see Table 2.1.

Steady state probabilities in an $M/M/\infty$ system

$$p_i = e^{-A}\frac{A^i}{i!},\ \forall i \in \mathbb{N} \tag{6.23}$$

To determine the stationary solution, an alternative approach is to partition the state space $\Omega = \{i\}_{i=0}^{k}$ and to use the *equilibrium equation* (5.40) to set up a system of linear equations. This principle is described in Section 5.2.4 and is now applied to the $M/M/\infty$-model.

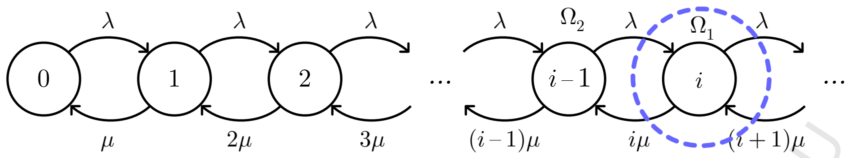


Figure 6.6: Node equations - partitioning the state space at states

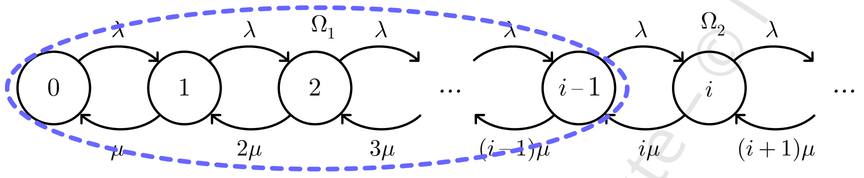


Figure 6.7: Cut equations - partitioning the state space between states

Node equations

A first method consists in partitioning the state space Ω in two disjoint state spaces $\Omega_1 = \{i\}$ and $\Omega_2 = \Omega - \Omega_1 = \{j\}_{j=0,\, j\neq i}^{n}$. This is illustrated in Figure 6.6.

The expected number of transitions from Ω_1 to Ω_2 over a long time period τ is $\tau p_i(\lambda + i\mu)$ and the expected number of transitions from Ω_2 to Ω_1 is $\tau p_{i-1}\lambda + \tau p_{i+1}(i+1)\mu$. Using the equilibrium equation, we obtain

$$p_i(\lambda + i\mu) = p_{i-1}\lambda + p_{i+1}(i+1)\mu,\ \forall i \in \mathbb{N}^* \tag{6.24}$$

which is exactly the same as (6.20).

Cut equations

A second method consists in partitioning the state space Ω in two disjoint state spaces $\Omega_1 = \{j\}_{j=0}^{i-1}$ and $\Omega_2 = \Omega - \Omega_1 = \{j\}_{j=i}^{k}$. This is illustrated in Figure 6.7.

The expected number of transitions from Ω_1 to Ω_2 over a long time period τ is $\tau p_{i-1}\lambda$ and the expected number of transitions from Ω_2 to Ω_1 is $\tau p_i i\mu$. Using the equilibrium equation, we obtain

$$p_{i-1}\lambda = p_i i\mu,\ \forall i \in \mathbb{N}^* \tag{6.25}$$

Rearranging and solving recursively (6.25) gives

$$p_i = p_0 \frac{A^i}{i!},\ \forall i \in \mathbb{N} \tag{6.26}$$

which is identical to (6.22).

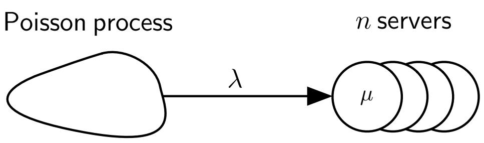


Figure 6.8: Erlang's loss model

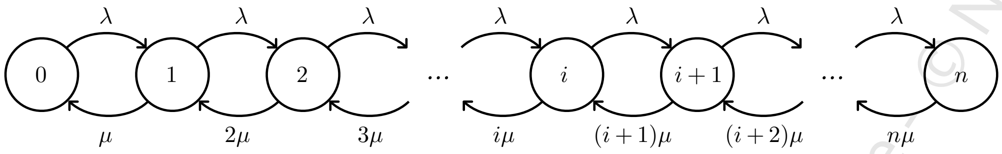


Figure 6.9: State transition diagram for the Erlang's loss model

Cut equations (6.25) are simpler than node equations (6.24) (only 2 state probabilities involved instead of 3) in the case of one-dimensional state transition diagrams.

6.3 Loss systems (M/M/n/n)

In this section, we consider systems with a finite number of servers and no queue ($M/M/n/n$-model). The number of states is therefore $n + 1$. Since there is no queue, the number of customers in the system is equal to the number of busy servers. In teletraffic theory, these models are referred to as *loss systems* because new requests are lost (blocked) upon arrival to the system if no server is idle. Depending on whether the number of sources is assumed to be *infinite* or *finite*, the $M/M/n/n$-model is called *Erlang's* or *Engset's loss model*, respectively. In the case of a finite number of sources (PCT-II traffic), the model is usually referred to as $M_x/M/n/n$ to indicate the state dependence of the arrival process. Only the stationary solutions are given in this section.

6.3.1 Erlang's loss model

The *Erlang*[1]'s *loss model* is the most commonly used model in teletraffic theory. An Erlang's loss model assumes PCT-I traffic with intensity λ, see Figure 6.8. The state transition diagram for such a system is given in Figure 6.9.

[1]Agner Krarup Erlang (1878-1929) was a Danish mathematician, statistician, and engineer who did pioneer work in the fields of traffic engineering and queueing theory.

Stationary solution

In the following, the service times are assumed to be negatively exponentially distributed with parameter μ, but the results provided in this section are proven to be valid for all types of renewal service processes. The offered traffic is $A = \lambda/\mu$, see (6.3) ($\mathrm{E}(X) = 1/\mu$).

The cut equations are the same as in the $M/M/\infty$ model, see Section 6.2. Therefore we determine the steady state probabilities from (6.22) but, as the number of states is limited, the normalization condition is now $\sum_{i=0}^{n} p_i = 1$ and we obtain

Steady state probabilities in an Erlang's loss system

$$p_i = \frac{A^i/i!}{\sum_{\nu=0}^{n} A^\nu/\nu!},\ i = 0, \cdots, n \tag{6.27}$$

This is the *truncated Poisson distribution* with parameter A.

Traffic characteristics

The *time congestion* $E_n(A)$, or $E_{1,n}(A)$, is the fraction of time all the servers are busy. The time congestion in Erlang's loss model is determined by the *Erlang's B-formula* which is valid for all renewal service processes [16].

Erlang's B-formula (time congestion in an Erlang's loss system)

$$E_n(A) = \frac{A^n/n!}{\sum_{\nu=0}^{n} A^\nu/\nu!} \tag{6.28}$$

This formula also has a recursive form under the assumption that $E_0(A) = 1$.

Recursive Erlang's B-formula

$$E_n(A) = \frac{AE_{n-1}(A)}{n + AE_{n-1}(A)} \tag{6.29}$$

The congestion values in Figure 6.10 are extremely high compared to what is normally acceptable, but the purpose of these plots is only to demonstrate how the time congestion changes as the traffic changes. The congestion increases as the offered traffic increases, and decreases as the number of servers increases. Note also that the time congestion is decreasing even when the offered traffic per server is constant ($n = A$). This is the effect of sharing resources (servers). Servers are grouped in a pool and shared among many; any server may serve any customer. If the servers were not shared but dedicated, the time congestion would have been constant for a constant offered traffic per server. The

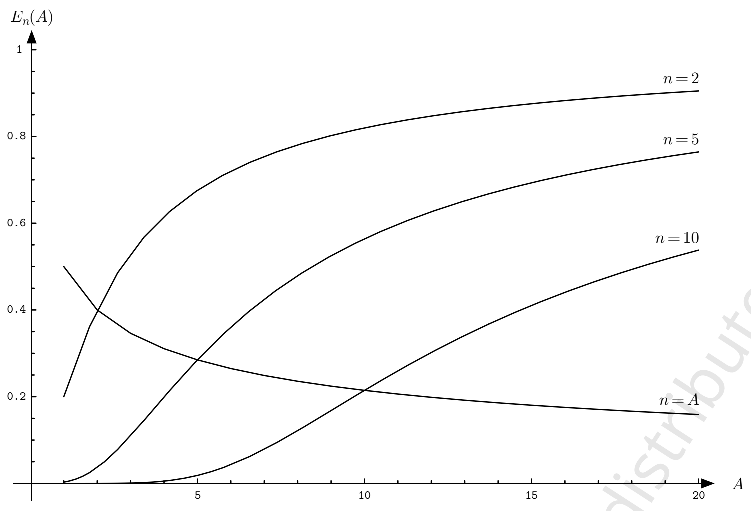


Figure 6.10: Time congestion as function of the offered traffic

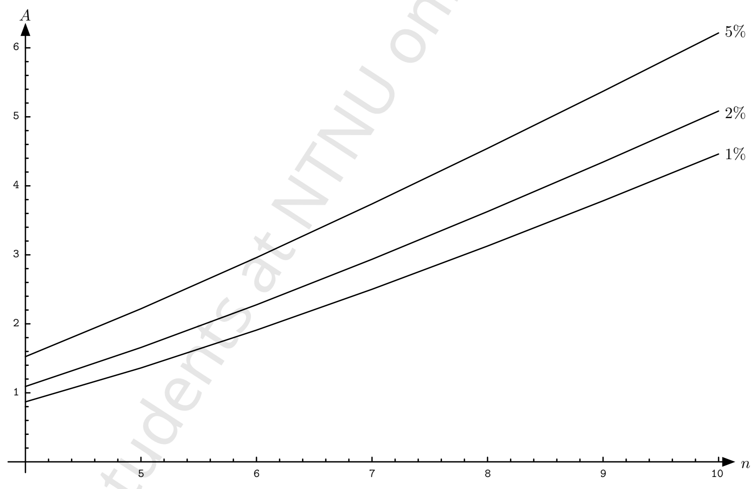


Figure 6.11: Offered traffic as a function of the number of servers

effect of resource sharing is also illustrated in Figure 6.11 by showing the offered traffic handled for fixed time congestions as the number of servers changes.

The *call congestion* $B_n(A)$ is the probability that an arrival is blocked. If we study the system over a long period of time τ, the expected time in state n is τp_n and the expected number of arrivals in state n is $\lambda\tau p_n$. The call congestion is equal to the number of arrivals in state n divided by the total number of arrivals in $(0,\tau)$.

Call congestion in an Erlang's loss system

$$B_n(A) = \frac{\lambda\tau p_n}{\sum_{i=0}^{n}\lambda\tau p_i} = \frac{p_n}{\sum_{i=0}^{n} p_i} = p_n = E_n(A) \tag{6.30}$$

The carried traffic is calculated using (6.8). As there is no queue, the number of customers in the system is equal to the number of busy servers, $\min(i,n) = i$, $\forall i$, and the maximum number of customers in the system to the number of servers, $k = n$. Hence

Carried traffic in an Erlang's loss system

$$A' = \sum_{i=0}^{n} i p_i = \sum_{i=1}^{n} \frac{\lambda}{\mu} p_{i-1} = \frac{\lambda}{\mu}\sum_{i=0}^{n-1} p_i = A(1 - E_n(A)) \tag{6.31}$$

Lost traffic in an Erlang's loss system

$$A'' = A - A' = AE_n(A) \tag{6.32}$$

6.3.2 Engset's loss model

An *Engset*[2]'s *loss model* [33] assumes PCT-II traffic, i.e. that the arrival intensity depends on the state of the system, $\lambda_i = (s-i)\gamma$. If the number of sources s is larger than the number of servers n $(s > n)$, then arrivals might get lost. The service times are assumed to be negatively exponentially distributed with parameter μ. As for the Erlang's loss model, this assumption may be relaxed, and the results provided in this section are proven to be valid for all types of renewal service processes. In the rest of this section, we assume $n \leqslant s$. The state transition diagram for such a system is given in Figure 6.12.

[2]Tore Olaus Engset (1865-1943) was a Norwegian mathematician, engineer and manager who did pioneer work in the field of telephone traffic queueing theory.

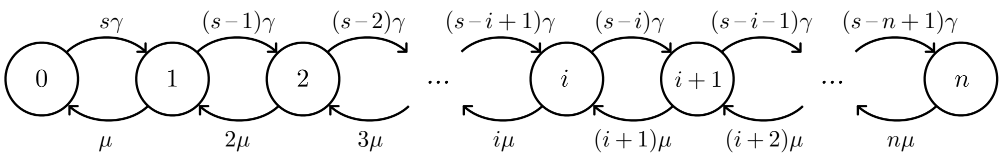


Figure 6.12: State transition diagram for Engset's loss model

Stationary solution

Under stationary conditions, the cut equations are

$$p_i i\mu = p_{i-1}(s-(i-1))\gamma, \ i = 1, \cdots, n \tag{6.33}$$

Rearranging and substituting $\beta = \gamma/\mu$, see (6.6) (E$(X) = 1/\mu$), gives

$$p_i = p_{i-1}\frac{s-(i-1)}{i}\beta \tag{6.34}$$

Then

$$p_i = p_0\frac{s}{1}\cdot\frac{s-1}{2}\cdots\frac{s-(i-1)}{i}\beta^i = p_0\frac{s!}{i!(s-i)!}\beta^i = p_0\binom{s}{i}\beta^i \tag{6.35}$$

Using the normalization condition $\sum_{i=0}^{n} p_i = 1$, we get

$$p_i = \frac{\binom{s}{i}\beta^i}{\sum_{\nu=0}^{n}\binom{s}{\nu}\beta^\nu}, \ i = 0, \cdots, n \tag{6.36}$$

which can be rewritten by substituting $\beta = \dfrac{\alpha}{1-\alpha}$.

$$\frac{\beta^i}{\sum_{\nu=0}^{n}\beta^\nu} = \frac{\left(\frac{\alpha}{1-\alpha}\right)^i}{\sum_{\nu=0}^{n}\left(\frac{\alpha}{1-\alpha}\right)^\nu} = \frac{\alpha^i\cdot\frac{(1-\alpha)^{s-i}}{(1-\alpha)^s}}{\sum_{\nu=0}^{n}\alpha^\nu\cdot\frac{(1-\alpha)^{s-\nu}}{(1-\alpha)^s}} = \frac{\alpha^i(1-\alpha)^{s-i}}{\sum_{\nu=0}^{n}\alpha^\nu(1-\alpha)^{s-\nu}} \tag{6.37}$$

Therefore

Steady state probabilities in an Engset's loss system

$$p_i = \frac{\binom{s}{i}\alpha^i(1-\alpha)^{s-i}}{\sum_{\nu=0}^{n}\binom{s}{\nu}\alpha^\nu(1-\alpha)^{s-\nu}}, \ i = 0, \cdots, n, \ s \geqslant n \tag{6.38}$$

When $n = s$ this is the binomial distribution with parameters α and s, see Table 2.1. When $s > n$ it is called a *truncated binomial distribution*

Traffic characteristics

The *time congestion* $E_{n,s}(\beta)$ is the fraction of time all servers are busy.

Time congestion in an Engset's loss system

$$E_{n,s}(\beta) = p_n = \frac{\binom{s}{n}\beta^n}{\sum_{\nu=0}^{n}\binom{s}{\nu}\beta^\nu}, \ s \geqslant n \tag{6.39}$$

The *call congestion* $B_{n,s}(\beta)$ is the probability that an arrival is blocked. If we study the system over a long period of time τ, the expected time in state n is τp_n and the expected number of arrivals in state n is $\gamma(s-n)\tau p_n$. The call congestion is equal to the number of arrivals in state n divided by the total number of arrivals in $(0,\tau)$.

$$B_{n,s}(\beta) = \frac{\gamma(s-n)\tau p_n}{\sum_{\nu=1}^{n}\gamma(s-\nu)\tau p_\nu} = \frac{\gamma(s-n)\tau\binom{s}{n}\beta^n}{\sum_{\nu=0}^{n}\gamma(s-\nu)\tau\binom{s}{\nu}\beta^\nu} = \frac{(s-n)\binom{s}{n}\beta^n}{\sum_{\nu=0}^{n}(s-\nu)\binom{s}{\nu}\beta^\nu} \tag{6.40}$$

Call congestion in an Engset's loss system

$$B_{n,s}(\beta) = \frac{(s-n)\binom{s}{n}\beta^n}{\sum_{\nu=0}^{n}(s-\nu)\binom{s}{\nu}\beta^\nu}, \ s \geqslant n \tag{6.41}$$

For $s = n$, $B_{n,s}(\beta) = 0$. For $s > n$, $(s-n)\begin{pmatrix} s \\ n \end{pmatrix} = s\begin{pmatrix} s-1 \\ n \end{pmatrix}$ and thus

$$B_{n,s}(\beta) = \frac{s\binom{s-1}{n}\beta^n}{\sum_{\nu=0}^{n}s\binom{s-1}{\nu}\beta^\nu} = \frac{\binom{s-1}{n}\beta^n}{\sum_{\nu=0}^{n}\binom{s-1}{\nu}\beta^\nu} = E_{n,s-1}(\beta) \tag{6.42}$$

The call congestion $B_{n,s}(\beta)$ for s sources and n servers is equal to the time congestion with one source less and the same number of servers $E_{n,s-1}(\beta)$. This result is known as the *arrival theorem*.

Arrival theorem

For a system with a limited number of sources, a service request (arrival) sees the system as if its source was not part of the system.

As $E_{n,s}(\beta)$ increases when s increases, $B_{n,s}(\beta) = E_{n,s-1}(\beta) < E_{n,s}(\beta)$.

The carried traffic is calculated using (6.8). As there is no queue, the number of customers in the system is equal to the number of busy servers, $\min(i,n) = i$, $\forall i$, and the maximum number of customers in the system to the number of servers, $k = n$. Hence

$$A' = \sum_{i=0}^{n} ip_i = \sum_{i=1}^{n} p_{i-1}(s-(i-1))\beta = \sum_{i=0}^{n-1} p_i(s-i)\beta \tag{6.43}$$

which simplifies to

$$A' = \sum_{i=0}^{n-1} p_i(s-i)\beta = \beta\left(\sum_{i=0}^{n-1} sp_i - \sum_{i=0}^{n-1} ip_i\right) = \beta(s(1-p_n) - (A' - np_n)) \tag{6.44}$$

Rearranging, we obtain

$$\begin{aligned} A'(1+\beta) = \beta(s - p_n(s-n)) \Leftrightarrow A' &= \frac{\beta}{\beta+1}(s - p_n(s-n)) \\ \Leftrightarrow A' &= \frac{\beta}{\beta+1}(s - E_{n,s}(\beta)\cdot(s-n)) \end{aligned} \tag{6.45}$$

Moreover, as $E_{n,s}(\beta) = \dfrac{s}{s-n}\dfrac{B_{n,s}(\beta)}{1+\beta(1-B_{n,s}(\beta))}$ [16], A' can be rewritten as

$$A' = \frac{s\beta(1-B_{n,s}(\beta))}{1+\beta(1-B_{n,s}(\beta))} \tag{6.46}$$

Carried traffic in an Engset's loss system

$$A' = \frac{\beta}{\beta+1}(s - E_{n,s}(\beta)\cdot(s-n)) = \frac{s\beta(1-B_{n,s}(\beta))}{1+\beta(1-B_{n,s}(\beta))} \tag{6.47}$$

The offered traffic is equal to the carried traffic when there is no loss, i.e. when the number of servers is equal to the number of sources, $n = s$. It is the expected value of the binomial distribution with parameters α and s, see Table 2.1. We find again (6.7)

$$A = \sum_{i=0}^{s} i\, p_i\big|_{n=s} = s\alpha = s\frac{\beta}{\beta+1} \tag{6.48}$$

Lost traffic in an Engset's loss system

$$A'' = A - A' = s\frac{\beta}{\beta+1} - \frac{\beta}{\beta+1}(s - E_{n,s}(\beta)\cdot(s-n)) = \frac{(s-n)\beta}{\beta+1}E_{n,s}(\beta) \tag{6.49}$$

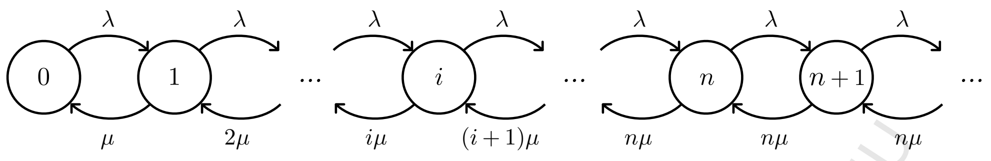


Figure 6.13: State transition diagram for the M/M/n model

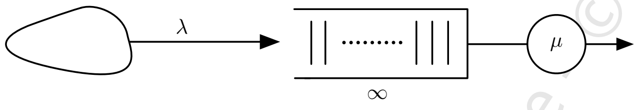


Figure 6.14: Single-server Erlang's delay system (M/M/1)

6.4 Erlang's delay systems (M/M/n)

In this section, we consider systems with n servers and an infinite number of queueing positions with Poisson arrivals with constant intensity λ (PCT-I traffic) and negative exponential services times with rate μ ($M/M/n$ systems). The state transition diagram of such a system is given in Figure 6.13. A customer is either being served or waiting in the queue. Contrary to loss systems presented in Section 6.3, the number of customers in a delay system $(0\ldots\infty)$ differs from the number of busy servers $(0\ldots n)$. In state $i,\ 0 \leqslant i \leqslant n$, there are i customers in the system, i servers are busy and $n-i$ servers are idle. In state $i,\ i \geqslant n$, there are i customers in the system, the n servers are busy and there are $i-n$ customers in the queue.

For the sake of simplicity, only single-server delay systems ($M/M/1$ model), see Figure 6.14, are addressed in details in this book. Results for $M/M/n$ systems can be obtained by following a similar approach and are given in appendix at the end of this chapter.

6.4.1 Stationary solution for an M/M/1 system

The state transition diagram for the $M/M/1$ model is given in Figure 6.15.

When the system is stationary, the cut equations are:

$$p_i\mu = p_{i-1}\lambda,\ i > 0 \tag{6.50}$$

Substituting the offered traffic $A = \lambda/\mu$, see (6.3) ($\mathrm{E}(X) = 1/\mu$), we obtain by recursion

$$p_i = p_0 A^i,\ i \geqslant 0 \tag{6.51}$$

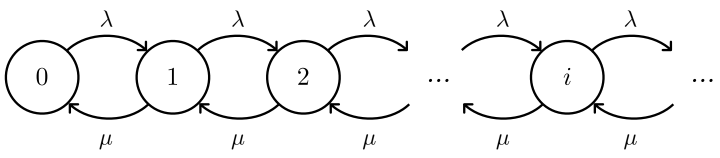


Figure 6.15: State transition diagram for the M/M/1 model

Using the normalization condition $\sum_{i=0}^{\infty} p_i = 1$, we obtain p_0.

$$p_0 \sum_{i=0}^{\infty} A^i = 1 \tag{6.52}$$

For the geometric series $\sum_{i=0}^{\infty} A^i$ to converge, the offered traffic A must be strictly smaller than 1. Statistical equilibrium is obtained only for $A < 1$. If $A \geqslant 1$, the queue grows to infinity.

Stability condition for an $M/M/1$ system

$$A < 1 \tag{6.53}$$

If $A < 1$, we obtain

$$p_0 = 1 - A \tag{6.54}$$

and

Steady state probabilities in an $M/M/1$ system

$$p_i = (1 - A)A^i, \ i \geqslant 0, \ A < 1 \tag{6.55}$$

This is the geometric distribution with parameter $(1 - A)$, see Table 2.1.

6.4.2 Traffic characteristics for an M/M/1 system

Since the number of queueing positions is infinite, no service requests are lost and the carried traffic is equal to the offered traffic when the system is stable ($A < 1$).

Carried traffic in an $M/M/1$ system

$$A' = \sum_{i=0}^{\infty} \min(i, 1) p_i = \sum_{i=1}^{\infty} 1 \cdot p_i = 1 - p_0 = A, \ A < 1 \tag{6.56}$$

If the system is unstable ($A \geqslant 1$) the carried traffic is $A' = 1$; the number of customers in the system tends to infinity and the server is always busy.

If the server is idle, a customer is served immediately upon arrival. The probability that a customer has to wait, $E_{2,1}(A) = P(W > 0)$, is the probability that it sees the server busy upon arrival. If we study the system over a long period of time τ, the expected time the system is in state i is τp_i and the expected number of arrivals in state i is $\lambda \tau p_i$. The probability that a customer has to wait is equal to the number of arrivals in any state $i > 0$ divided by the total number of arrivals in $(0, \tau)$.

$$E_{2,1}(A) = \frac{\sum_{i=1}^{\infty} \lambda \tau p_i}{\sum_{i=0}^{\infty} \lambda \tau p_i} = \sum_{i=1}^{\infty} p_i = 1 - p_0 = A,\ A < 1$$

$$\Leftrightarrow E_{2,1}(A) = \frac{\sum_{i=1}^{\infty} \lambda \tau p_i}{\sum_{i=0}^{\infty} \lambda \tau p_i} = \sum_{i=1}^{\infty} p_i = (1-A)A \sum_{i=1}^{\infty} A^{i-1} = p_1 \frac{1}{1-A} = A,\ A < 1 \tag{6.57}$$

This is the *Erlang's C-formula* for an $M/M/1$ system.

Erlang's C-formula for an $M/M/1$ system (probability of waiting)

$$E_{2,1}(A) = P(W > 0) = A,\ A < 1 \tag{6.58}$$

The probability that customers are waiting in the queue, i.e. that the queue length Q is strictly larger than 0, is the probability that the system is in any state $i > 1$.

$$P(Q > 0) = \sum_{i=2}^{\infty} p_i = A \sum_{i=2}^{\infty} p_{i-1} = A \sum_{i=1}^{\infty} p_i = A E_{2,1}(A) = A^2,\ A < 1 \tag{6.59}$$

Probability of having customers in queue in an $M/M/1$ system

$$P(Q > 0) = A E_{2,1}(A) = A^2,\ A < 1 \tag{6.60}$$

6.4.3 Expected numbers in an M/M/1 system

The expected number of customers in the system under stable conditions ($A < 1$) is

$$\mathrm{E}(I) = \sum_{i=1}^{\infty} i p_i = (1-A) \sum_{i=1}^{\infty} i A^i = (1-A)A \sum_{i=1}^{\infty} i A^{i-1},\ A < 1 \tag{6.61}$$

This can solved using that if $x < 1$ then $\sum i x^{i-1} = \frac{\mathrm{d}}{\mathrm{d}x} \sum x^i$.

$$\mathrm{E}(I) = (1-A)A\frac{\mathrm{d}}{\mathrm{d}A}\left(\sum_{i=1}^{\infty} A^i\right) = \frac{(1-A)A}{(1-A)^2} = \frac{A}{1-A},\ A<1 \tag{6.62}$$

This is this expected value of the geometric distribution with parameter $(1-A)$, see Table 2.1.

Expected number of customers in an $M/M/1$ system

$$\mathrm{E}(I) = \frac{A}{1-A},\ A<1 \tag{6.63}$$

The expected number of customers in queue, or expected queue length, under stable conditions $(A<1)$ is

$$\mathrm{E}(Q) = \sum_{i=2}^{\infty}(i-1)p_i = A\sum_{i=2}^{\infty}(i-1)p_{i-1} = A\sum_{i=1}^{\infty} ip_i = A\mathrm{E}(I) = \frac{A^2}{1-A},\ A<1 \tag{6.64}$$

The expected number of customers in queue is also equal to the expected number of customers in the system minus the expected number of customers being served. The expected number of customers being served is equal to the expected number of busy servers, that is to the carried traffic. We find again

$$\mathrm{E}(Q) = \mathrm{E}(I) - A = \frac{A^2}{1-A},\ A<1 \tag{6.65}$$

Expected queue length in an $M/M/1$ system

$$\mathrm{E}(Q) = \frac{A^2}{1-A},\ A<1 \tag{6.66}$$

The expected queue length given that the queue is not empty is found using the law of total expectation (2.57)

$$\mathrm{E}(Q) = \mathrm{E}(Q \mid Q>0)P(Q>0) + \underbrace{\mathrm{E}(Q \mid Q=0)P(Q=0)}_{=0} \tag{6.67}$$

Expected queue length in an $M/M/1$ system given that the queue is not empty

$$\mathrm{E}(Q \mid Q>0) = \frac{\mathrm{E}(Q)}{P(Q>0)} = \frac{1}{1-A},\ A<1 \tag{6.68}$$

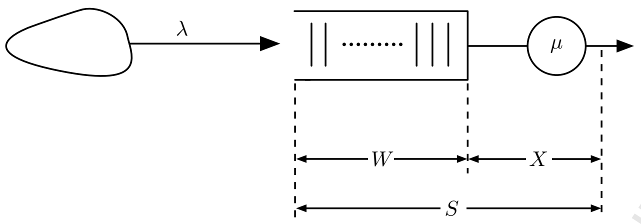


Figure 6.16: Times in queue (W), in service (X) and in system (S)

6.4.4 Expected times in an M/M/1 system

The sojourn time in the system S, the queueing (or waiting) time and the service time X have been defined in Section 6.1.1. Figure 6.16 gives an illustration in the case of an M/M/1 system.

The expected sojourn time in the system is determined by applying Little's formula to the entire system.

Expected sojourn time in an $M/M/1$ system

$$\mathrm{E}(S) = \frac{\mathrm{E}(I)}{\lambda} = \frac{1}{\lambda}\frac{A}{1-A} = \frac{1}{\mu(1-A)} = \frac{1}{\mu-\lambda},\ A<1 \tag{6.69}$$

If $A \to 0$ then $\mathrm{E}(S) \to \mathrm{E}(X)$. If $A \to 1$ then $\mathrm{E}(S) \to \infty$

Figure 6.17 shows the ratio between expected sojourn time in system $\mathrm{E}(S)$ and the expected service time $\mathrm{E}(X)$ as a function of offered traffic A. When the offered traffic exceeds 0.80 Erlang, the expected sojourn time in the system increases rapidly (the "knee" on the curve). Hence, when dimensioning a queueing system one should be on the safe side of the "knee"; the average offered traffic should be at most $0.50-0.60$ Erlang. Beyond these values, even small changes in the offered traffic cause significant changes in the sojourn time.

The expected queueing, or waiting, time is the expected sojourn time in the system minus the expected service time

Expected waiting time for all customers in an $M/M/1$ system

$$\mathrm{E}(W) = \mathrm{E}(S) - \mathrm{E}(X) = \frac{A}{\mu(1-A)} = \frac{A}{\mu-\lambda} = A\mathrm{E}(S),\ A<1 \tag{6.70}$$

The same result can be obtained by applying Little's formula to the queue.

$$\mathrm{E}(W) = \frac{\mathrm{E}(Q)}{\lambda} = \frac{A^2}{(1-A)\lambda} = \frac{A}{(1-A)\mu} = \frac{A}{\mu-\lambda} = A\mathrm{E}(S),\ A<1 \tag{6.71}$$

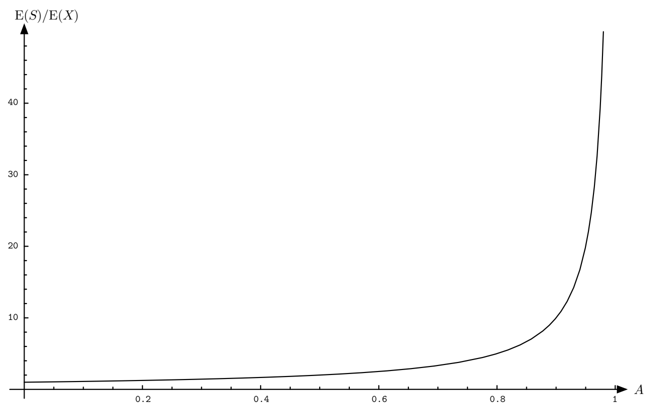


Figure 6.17: Ratio between expected sojourn time in system $\mathrm{E}(S)$ and the expected service time $\mathrm{E}(X)$ as a function of offered traffic A.

If $A \to 0$ then $\mathrm{E}(W) \to 0$. If $A \to 1$ then $\mathrm{E}(W) \to \infty$.

The expected waiting time given that the customer has to wait (delayed customer) $\mathrm{E}(W \mid W > 0)$ is found using the law of total expectation (2.57)

$$\mathrm{E}(W) = \mathrm{E}(W \mid W > 0)P(W > 0) + \underbrace{\mathrm{E}(W \mid W = 0)P(W = 0)}_{=0} \tag{6.72}$$

The last term is zero because if the customer is not delayed the waiting time is obviously zero. Then, rearranging and substituting gives

Expected waiting time for delayed customers in an $M/M/1$ system

$$\mathrm{E}(W \mid W > 0) = \frac{\mathrm{E}(W)}{E_{2,1}(A)} = \frac{\mathrm{E}(X)}{1-A} = \mathrm{E}(S) = \frac{1}{\mu - \lambda},\ A < 1 \tag{6.73}$$

6.4.5 Time distributions for an M/M/1/∞/FIFO system

As shown in previous section, the expected times in the system can be obtained from the expected numbers in the system using Little's formula (3.31). No assumption on the queueing discipline is required because only the expected values are considered. However, the time distributions depend on the queueing discipline. In this section, we consider the simplest case of FIFO queueing.

Waiting time distributions

Assume that the system is in state $i = q + 1$, where $q \in \mathbb{N}$ is the number of customers in queue immediately before a customer enters the system. This customer has to wait $q+1$ service completions before being served. When the server is busy, the system completes customers with a constant rate μ (negatively exponentially distributed service times), i.e. the departure process $N(t)$ is a Poisson process with intensity μ. Therefore, the waiting time for a customer that has to wait (delayed customer) given that there are q customers ahead in the queue is Erlang-$(q + 1)$ distributed.

$$\begin{aligned} F_{W|W>0}(t \mid q) &= P(W \leqslant t \mid q \text{ customers in the queue upon arrival}) \\ &= P(N(t) \geqslant q + 1) \end{aligned} \tag{6.74}$$

where $N(t)$ is the number of service completions in the interval $(0, t]$, see Section 3.1.2.

Hence,

Conditional waiting time distribution for delayed customers in an $M/M/1/\infty/FIFO$ system

Given that the server is busy and that there are $q \in \mathbb{N}$ customers in queue immediately before a new customer enters the system, the waiting time distribution for this customer in an $M/M/1/\infty/FIFO$ system is Erlang-$(q + 1)$ distributed with parameter μ.

$$F_{W|W>0}(t \mid q) = \sum_{j=q+1}^{\infty} \frac{(\mu t)^j}{j!} e^{-\mu t},\ q \in \mathbb{N},\ t > 0 \tag{6.75}$$

The unconditional waiting time distribution for delayed customers is obtained using the law of total probability (2.9).

$$F_{W|W>0}(t) = \sum_{q=0}^{\infty} F_{W|W>0}(t \mid q) p_{W>0_q} \tag{6.76}$$

where $p_{W>0_q}$ is the probability that there are q customers in the queue, i.e. $q+1$ customers in the system, immediately before a new customer enters the system given that the server is busy, i.e. given that this new customer has to wait. Under the condition that the arrival and observer distributions are identical, which holds when the arrival process is a Poisson process (*PASTA property*, see Section 8.1.2), and that $A = \lambda/\mu < 1$, we have

$$p_{W>0_q} = \frac{p_{q+1}}{E_{2,1}(A)} = \frac{(1 - A)A^{q+1}}{A} = (1 - A)A^q,\ q \in \mathbb{N},\ A < 1 \tag{6.77}$$

Thus (6.76) becomes

$$F_{W|W>0}(t) = \sum_{q=0}^{\infty} \sum_{j=q+1}^{\infty} \frac{(\mu t)^j}{j!} e^{-\mu t}(1 - A)A^q,\ t > 0,\ A < 1 \tag{6.78}$$

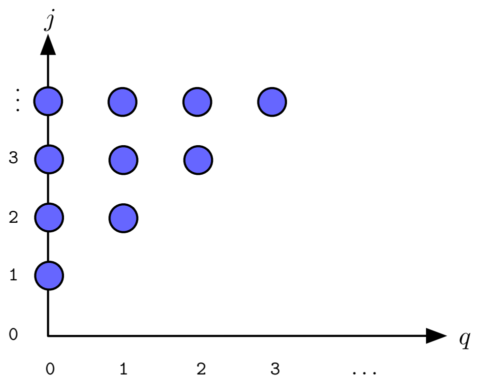


Figure 6.18: Interchanging the order of summation

In order to solve (6.78) we interchange the order of summation. See Figure (6.18) for help in interchanging j and q.

$$\sum_{q=0}^{\infty}\sum_{j=q+1}^{\infty} = \sum_{j=1}^{\infty}\sum_{q=0}^{j-1} \tag{6.79}$$

Then (6.78) becomes

$$F_{W|W>0}(t) = (1-A)\sum_{j=1}^{\infty}\frac{(\mu t)^j}{j!}e^{-\mu t}\sum_{q=0}^{j-1}A^q = \sum_{j=1}^{\infty}\frac{(\mu t)^j}{j!}e^{-\mu t}(1-A^j),\ t>0,\ A<1 \tag{6.80}$$

because $\sum_{q=0}^{j-1}A^q = \frac{1-A^j}{1-A}$.

Substituting $A=\lambda/\mu$ and since $1-A^0=0$, (6.80) can be further simplified:

$$\begin{aligned} F_{W|W>0}(t) &= \sum_{j=0}^{\infty}\frac{(\mu t)^j}{j!}e^{-\mu t} - \sum_{j=0}^{\infty}\frac{(\mu t)^j}{j!}e^{-\mu t}\left(\frac{\lambda}{\mu}\right)^j \\ &= 1 - e^{-\mu t}\sum_{j=0}^{\infty}\frac{(\lambda t)^j}{j!} \\ &= 1 - e^{-(\mu-\lambda)t},\ t>0,\ \lambda<\mu \end{aligned} \tag{6.81}$$

Hence,

Waiting time distribution for delayed customers in an $M/M/1/\infty/FIFO$ system

The waiting time for delayed customers in an $M/M/1/\infty/FIFO$ system is negatively exponentially distributed with parameter $(\mu - \lambda),\ \lambda < \mu$.

$$F_{W|W>0}(t) \sim \text{n.e.d.}(\mu - \lambda),\ t > 0,\ \lambda < \mu \tag{6.82}$$

The expected value of this distribution is $\dfrac{1}{\mu - \lambda}$ which is in agreement with (6.73).

To sum up, if a customer counts the number q of customers in the queue when entering the system the waiting time is Erlang-$(q+1)$ distributed with parameter μ, while if not the waiting time is negatively exponentially distributed with parameter $(\lambda - \mu)$.

Now, using (2.70), we have

$$F_{W|W>0}(t \mid 0) = \frac{F_W(t) - F_W(0)}{1 - F_W(0)} \tag{6.83}$$

and $F_W(0) = P(W \leqslant 0) = 1 - E_{2,1}(A),\ A < 1$.

We obtain the waiting time distribution for all customers:

Waiting time distribution for all customers in an $M/M/1/\infty/FIFO$ system

$$F_W(t) = 1 - E_{2,1}(A)e^{-(1-A)\mu t} = 1 - \frac{\lambda}{\mu}e^{-(\mu-\lambda)t},\ t \geqslant 0,\ \lambda < \mu \tag{6.84}$$

The expected value of this distribution is $\dfrac{E_{2,1}(A)}{(1-A)\mu}$ which is in agreement with (6.70).

Sojourn time distribution

Following a similar approach as for the waiting time distribution, if an $M/M/1$ system with FIFO queueing is in state i immediately before a customer enters the system, the sojourn time for this customer is Erlang-$(i+1)$ distributed, and the sojourn time, or response time, for an arbitrary customer is negatively exponentially distributed with parameter $(\mu - \lambda),\ \lambda < \mu$.

Sojourn time distribution in an $M/M/1/\infty/FIFO$ system

The sojourn time in an $M/M/1/\infty/FIFO$ system is negatively exponentially distributed with parameter $(\mu - \lambda),\ \lambda < \mu$.

$$F_S(t) \sim \text{n.e.d.}(\mu - \lambda),\ t \geqslant 0,\ \lambda < \mu \tag{6.85}$$

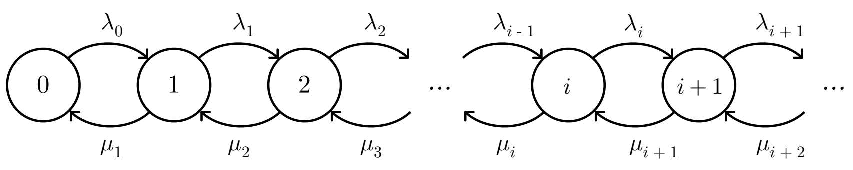


Figure 6.19: State transition diagram for a birth-death process

The expected value of this distribution is $\dfrac{1}{\mu - \lambda}$ which is in agreement with (6.69).

6.5 Stationary solution for birth-death processes

As mentioned in the introduction to this chapter, all the processes presented in this chapter are *birth-death processes*, that is processes which state may only change by ± 1 every time an event occurs. The generic state transition diagram for such a process is shown in Figure 6.19. λ_i denotes the arrival (birth) rate in state i and μ_i the departure (death) rate from state i.

Under stationary conditions, the cut equations are

$$p_i \lambda_i = p_{i+1} \mu_{i+1}, \; i = 0, \ldots, k-1 \tag{6.86}$$

where k denotes the maximum number of customers in the system. By recursion, we obtain

$$p_{i+1} = p_0 \prod_{j=0}^{i} \frac{\lambda_j}{\mu_{j+1}}, \; i = 0, \ldots, k-1 \tag{6.87}$$

Using the normalization condition $\sum_{i=0}^{k} p_i = 1$, we have

$$p_0 + \sum_{\nu=0}^{k-1} p_{\nu+1} = 1 \implies p_0 = \frac{1}{1 + \sum_{\nu=0}^{k-1} \prod_{j=0}^{\nu} \frac{\lambda_j}{\mu_{j+1}}} \tag{6.88}$$

Hence the following general result

Steady state probabilities for a birth-death process

$$p_{i+1} = \frac{\prod_{j=0}^{i} \frac{\lambda_j}{\mu_{j+1}}}{1 + \sum_{\nu=0}^{k-1} \prod_{j=0}^{\nu} \frac{\lambda_j}{\mu_{j+1}}}, \; i = 0, \ldots, k-1 \tag{6.89}$$

Using (6.89) and setting λ_i, μ_i and k to their respective values, we find again the results obtained previously; (6.23) for M/M/∞ systems, (6.27) for Erlang's loss systems, (6.38) for Engset's loss systems, and (6.93) for Erlang's delay systems.

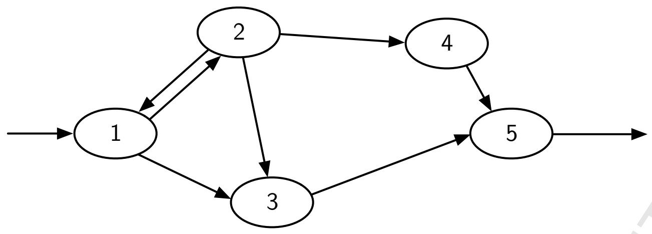


Figure 6.20: Jackson network with $k = 5$ nodes

6.6 Queueing networks

Each node in a communication network can be viewed and modeled as a queueing system. In this section we address *Jackson networks* which are arbitrary networks of $M/M/n$-queues.

Jackson [34] addressed modeling of an arbitrary network (cycles are allowed) of K nodes. The following assumptions apply:

- Customers arrive from outside the system at node k according to a Poisson process with intensity λ_k.
- The number of servers in node k is $n_k \geqslant 1$ where $k = 1, \cdots, K$. The service time in node k is negative exponentially distributed with parameter μ_k.
- Each node has an infinite number of queueing positions.
- After a service completion at node k a customer is either instantaneously routed to and received at node j with probability p_{kj}, or leaves instantaneously the system with probability $p_{k0} = 1 - \sum_{j=1}^{K} p_{kj}$.

The *total arrival intensity*, Λ_k, of customers to node k is the sum of the intensity λ_k of customers from outside the network and the intensities of customers routed from other nodes, $p_{jk}\Lambda_j$, $j = 1, \cdots, K$,

$$\Lambda_k = \lambda_k + \sum_{j=1}^{K} p_{jk}\Lambda_j, \; k = 1, \cdots, K \tag{6.90}$$

An example of a network with $K = 5$ nodes is shown in Figure (6.20) and a general network node is given in Figure (6.21).

Now, let $I_k = i_k$ be the number of customers in node k. The state of the network is then defined as the vector $(i_1, i_2, \cdots, i_K)$.

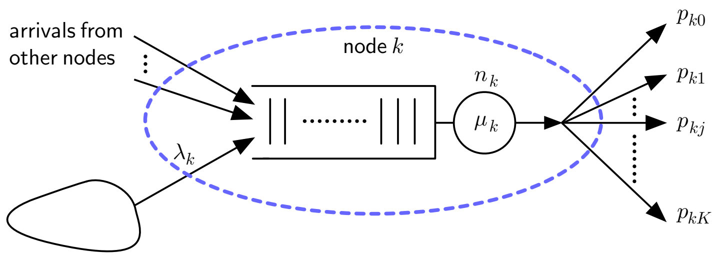


Figure 6.21: General model of a node in a Jackson network

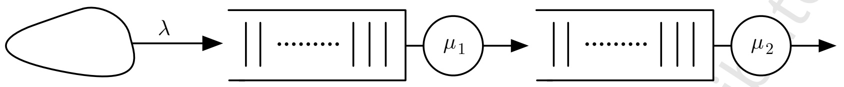


Figure 6.22: Tandem queue with 2 nodes

Then, the following theorem is true [34].

Jackson's theorem

Under steady state conditions ($\Lambda_k < n_k\mu_k,\ \forall k \in \{1, \dots, K\}$), the state probabilities of a Jackson network are equal to the product of the state probabilities of each individual node ($M/M/n_k$ queue).

$$P(i_1, i_2, \cdots, i_K) = P(I_1 = i_1, I_2 = i_2, \cdots, I_K = i_K) = \prod_{k=1}^{K} p(I_k = i_k) \qquad (6.91)$$

The Jackson's theorem implies that each individual node can be studied independently as long as the Λ_k is determined for each node. The proof of the *product form* in Jackson's theorem is given in [34]. The independence between the nodes is explained by *Burke's theorem* [35] which says that the departure process of an $M/M/n$-system is a Poisson process.

Example 6.4 Tandem queues
An illustration of two queues in tandem (series) is given in Figure 6.22. The offered traffic (arrivals from outside) to the first queue is $A_1 = \lambda/\mu_1$. If $A_1 < 1$ then the first queue is stable and the offered traffic to the second queue is $A_2 = \lambda/\mu_2$. If $A_2 < 1$ then the second queue is also stable. The queue with highest offered traffic is the bottleneck.

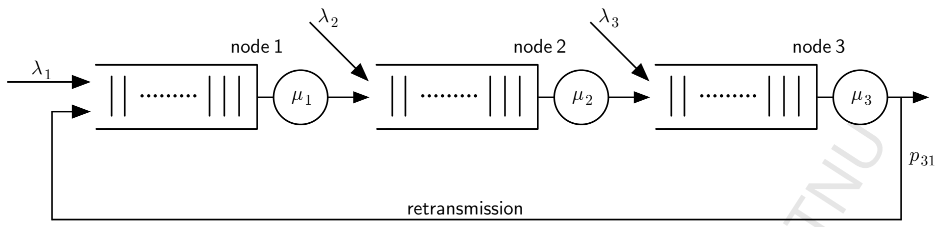


Figure 6.23: Feedback network

Example 6.5 Network with feedback
An illustration of a network with feedback is given in Figure 6.23. Node k ($k = 1, 2, 3$) represents a transmission link with a buffer in front of the link. If the capacity of link k is c_k and the packet length is B, then the expected service time is $1/\mu_k = \mathrm{E}(B)/c_k$. The packets are retransmitted from node 1 with probability p_{31}. The total arrival intensity to each node can be determined using (6.90).

$$\begin{cases} \Lambda_1 = \lambda_1 + p_{31}\Lambda_3 \\ \Lambda_2 = \lambda_2 + \Lambda_1 \\ \Lambda_3 = \lambda_3 + \Lambda_2 \end{cases}$$

Solving this system of equations gives

$$\begin{cases} \Lambda_1 = \dfrac{\lambda_1 + p_{31}(\lambda_2 + \lambda_3)}{1 - p_{31}} \\ \Lambda_2 = \dfrac{\lambda_1 + \lambda_2 + p_{31}\lambda_3}{1 - p_{31}} \\ \Lambda_3 = \dfrac{\lambda_1 + \lambda_2 + \lambda_3}{1 - p_{31}} \end{cases}$$

The expected number of packets, $\mathrm{E}(I_k)$, in node k and the expected sojourn time, $\mathrm{E}(S_k)$, in node k are determined using (6.63) and (6.69), respectively, with $\lambda = \Lambda_k$, $\mu = \mu_k$ and $A = \Lambda_k/\mu_k$.
The expected total system time for arrivals to node 1 is

$$\mathrm{E}(S) = (\mathrm{E}(R) + 1)\sum_{k=1}^{3} \mathrm{E}(S_k)$$

where $\mathrm{E}(R)$ is the expected number of retransmissions.
R is geometrically distributed $f_R(r) = p_{31}^r(1-p_{31})$. The expected value of R is $\mathrm{E}(R) = p_{31}/(1-p_{31})$.

In this section, we have only considered networks of $M/M/n$ queues. Other types of nodes give also simple results, see [16] and [17] and the references therein for further details. For instance, nodes with an infinite number of servers and a general renewal service process can be used to model a delay independent of the traffic.

Appendix: Results for M/M/n systems

Results for $M/M/n$ systems can be obtained following the same line of reasoning as for $M/M/1$ systems. The details of the calculations are beyond the scope of this book, see for instance [16]. By setting $n = 1$, we find again the results obtained in Section 6.4.

Stability condition for an $M/M/n$ system

$$A < n \tag{6.92}$$

Steady state probabilities in an $M/M/n$ system

$$p_i = \begin{cases} \dfrac{\frac{A^i}{i!}}{\sum_{i=0}^{n-1} \frac{A^i}{i!} + \frac{A^n}{n!}\frac{n}{n-A}}, & 0 \leqslant i \leqslant n \\ \dfrac{\frac{A^i}{n! n^{i-n}}}{\sum_{i=0}^{n-1} \frac{A^i}{i!} + \frac{A^n}{n!}\frac{n}{n-A}}, & i \geqslant n \end{cases}, \; A < n \tag{6.93}$$

Carried traffic in an $M/M/n$ system

$$A' = A, \; A < n \tag{6.94}$$

Erlang's C-formula (probability of waiting in an $M/M/n$ system)

$$E_{2,n}(A) = P(W > 0) = p_n \frac{n}{n-A} = \frac{\frac{A^n}{n!}\frac{n}{n-A}}{\sum_{i=0}^{n-1} \frac{A^i}{i!} + \frac{A^n}{n!}\frac{n}{n-A}}, \; A < n \tag{6.95}$$

Probability of having customers in queue in an $M/M/n$ system

$$P(Q > 0) = \frac{A}{n} E_{2,n}(A), \; A < n \tag{6.96}$$

Expected number of customers in an $M/M/n$ system

$$\mathrm{E}(I) = \frac{A}{n-A} \left(E_{2,n}(A) + n - A\right), \; A < n \tag{6.97}$$

Expected queue length in an $M/M/n$ system

$$\mathrm{E}(Q) = \frac{A}{n-A} E_{2,n}(A),\ A < n \tag{6.98}$$

Expected queue length in an $M/M/n$ system given that the queue is not empty

$$\mathrm{E}(Q \mid Q > 0) = \frac{\mathrm{E}(Q)}{P(Q > 0)} = \frac{n}{n-A},\ A < n \tag{6.99}$$

Expected sojourn time in an $M/M/n$ system

$$\mathrm{E}(S) = \frac{1}{(n-A)\mu} \left(E_{2,n}(A) + n - A\right),\ A < n \tag{6.100}$$

Expected waiting time for all customers in an $M/M/n$ system

$$\mathrm{E}(W) = \frac{1}{\mu(n-A)} E_{2,n}(A),\ A < n \tag{6.101}$$

Expected waiting time for delayed customers in an $M/M/n$ system

$$\mathrm{E}(W \mid W > 0) = \frac{\mathrm{E}(W)}{E_{2,n}(A)} = \frac{1}{\mu(n-A)},\ A < n \tag{6.102}$$

Conditional waiting time distribution for delayed customers in an $M/M/n/\infty/FIFO$ system

Given that all servers are busy and that there are $q \in \mathbb{N}$ customers in queue immediately before a new customer enters the system, the waiting time distribution for this customer in an $M/M/n/\infty/FIFO$ system is Erlang-$(q+1)$ distributed with parameter $n\mu$.

$$F_{W|W>0}(t \mid q) = \sum_{j=q+1}^{\infty} \frac{(\mu t)^j}{j!} e^{-\mu t},\ q \in \mathbb{N},\ t > 0 \tag{6.103}$$

Waiting time distribution for delayed customers in an $M/M/n/\infty/FIFO$ system

The waiting time for delayed customers in an $M/M/n/\infty/FIFO$ system is negatively exponentially distributed with parameter $(n\mu - \lambda),\ \lambda < n\mu$.

$$F_{W|W>0}(t) \sim \text{n.e.d.}(n\mu - \lambda),\ t > 0,\ \lambda < n\mu \tag{6.104}$$

Waiting time distribution for all customers in an $M/M/n/\infty/FIFO$ system

$$F_W(t) = 1 - E_{2,n}(A)e^{-(n-A)\mu t},\ t \geqslant 0,\ A < n \tag{6.105}$$

7 Dependability models

In this chapter we take a closer look at basic techniques for dependability modeling and analysis. In order to make the models analytically tractable, assumptions are required. For instance:

- Markov properties, see Section 3.3.1, must be assumed to use state transition diagrams for quantitative analysis, and
- independence, see Section 2.1.4, between failures and repairs of the system components must be assumed to use structural models for quantitative analysis.

It is important to understand what these assumptions imply and the limitations they cause. Note also that, although it is only required when using state transition diagrams, it is commonly assumed that failures occur according to a Poisson process. Furthermore, it is also important

- to consider all the relevant failures and understand which underlying process causes which failure. For instance, failures due to logical faults, e.g. in the software, were for too long neglected as a significant contributor to the lack dependability in ICT-systems.
- to have insight in the mechanisms applied to detect and tolerate failures, and the mechanisms and countermeasures taken to repair the failures. Dependability models must capture the correct behavior of these mechanisms, but also the fact that fault handling mechanisms are not necessarily perfect and may fail.
- to understand how failure may propagate between cooperating components.

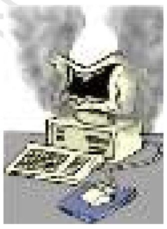
The purpose of this chapter is not to provide a comprehensive discussion on all the aspects mentioned above. Rather, it gives a brief introduction to dependability modeling in Section 7.1 and presents two dependability modeling approaches, namely Markov models in Section 7.2 and structural models in Section 7.3. Finally, Section 7.4 briefly discusses how to combine these two approaches.

7.1 Basics

This section briefly introduces the basics for developing dependability models and designing dependable systems. For a comprehensive presentation, see e.g. [9, 36, 37, 38], and for repetition of the basic concepts, see Section 1.2.2. For a thorough introduction to the dependability concepts, [14] is recommended.

7.1.1 Types of faults

Physical faults



Physical faults, or solid faults, are the "classical" faults, e.g. wear out of components, random device defects due to manufacturing imperfections, physical degradation of the material of the component as a result of electrical overstress, pollution, mechanical stress or shock. Physical faults are internal to the system. They are permanent, that is, as soon as a physical fault has occurred it persists until repaired. They occur in the operational phase of the life cycle of system, both during use and inactivity (in store, serving as a back up or simply powered off).

Transient faults

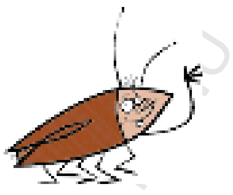
Transient faults are present only for a short period of time and no physical change occurs in the system. Transient faults are usually caused by disturbances external to the system itself like electromagnetic interference and radiation. Typical transient faults are noise on a transmission line and alpha-particles passing through RAM flipping bits. Transient faults occur during the operational phase of the life cycle of a system and have a physical origin. In practice, it may be very difficult to distinguish between transient and intermittent faults.

Intermittent faults

Intermittent faults, or sporadic faults, come and go. They may be due to changes of the parameters of a hardware component caused by temperature, aging etc., marginal timing, synchronization in hardware or software, specific input patterns, etc. Intermittent faults may have a physical or human origin. They are problematic to deal with since they are very difficult to localize and correct. Many dependability predictions take only permanent physical faults into consideration. There are several investigations showing that temporary physical faults (transient and intermittent) are far more frequent e.g. [39].

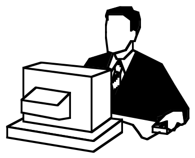
Design faults

Design, or logical, faults are human made faults which affect the logic of a system. This class of faults spans from basic design flaws to trivial implementation failures in the circuitry and software in a system. Lack of proper timing and synchronization are other examples of design faults. In this context, design faults also encompass specification faults, e.g. inconsistent system specification. Design faults may be found both in hardware and software, as well as in the interaction between hardware and software. Since most of the logic of an ICT-system is specified in software, most of the design faults are also associated with the software.



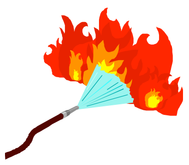
Software faults are a subgroup of the design faults and are popularly called bugs. We distinguish Heisenbugs and Bohrbugs. Heisenbugs are software faults which cause failures that are almost impossible to reproduce. For instance, a slight change in the timing of the input values or in the overall system state may change the course of processing so much that a failure cannot be reproduced. Bohrbugs are software faults which always cause the same failure if the conditions are roughly the same, i.e. reproducible failures. The mechanisms behind software failing and tolerance, and the significance of the distinction between Heisenbugs and Bohrbugs are discussed in [9].

Interaction or operational faults



Interaction or operational faults are accidental faults made by humans operating or maintaining a system. Examples includes misconfiguration of the system, the repairman who accidentally shuts down and starts repairing the working computer in a duplicated system instead of the failed, or the unix (root) user who accidentally types `rm -r * xyz` instead of `rm -r *.xyz`. As fault-tolerance is introduced to avoid that the previous described types of faults result in a system failure, fool-tolerance may be introduced to prevent failures from interaction faults. In addition to pure accidents and sloppiness, interaction faults may also be due to poor or incorrect documentation, inadequate training, poor or misleading human-machine interface etc.

Faults caused by the environment



An ICT-system fails if it gets no power, it is submerged into water due to flood, the building it is in burns down, etc. These failures of the environment cause the system to fail and thus reduce the dependability of the service provided.

Some statistics of the effect of the fault types on system performance

Examples of measurements of failures in operational systems are shown in the figures below.

Figure 7.1 shows the relative impact of the different types of faults on the system failure intensity of Tandem Computers[1] server systems. These observations are relatively old,

[1]Through several acquisitions, the successors of these system are now (2007) delivered by HP.

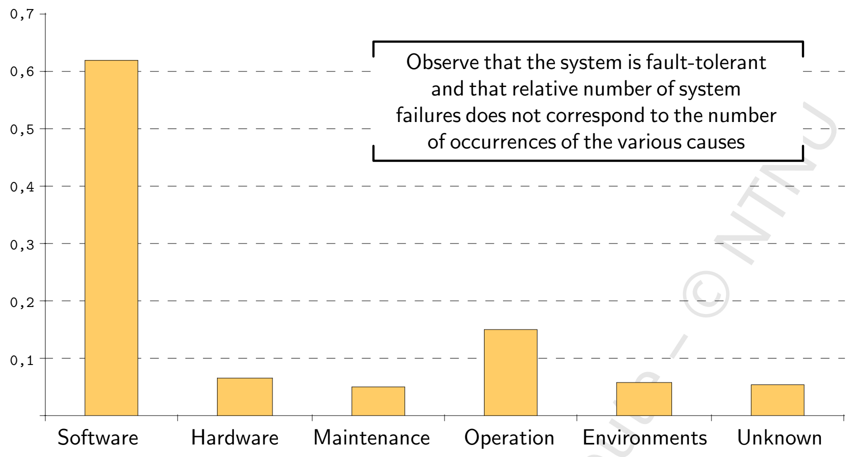


Figure 7.1: Causes of system failures in Tandem Computer systems in 1989 [40]

but serve to illustrate that none of the types of faults should be neglected and the impact of software/logical faults may be significant.

Figure 7.2 shows the effect of different types of faults in terms of major outages in the US public switched telephone network. A major outage in this context, is the loss of service of at least 30 000 subscribers lasting for some minutes. Observe that the failures of the fault management software contribute more than the software providing the telephony service itself.

Note that both of these figures show failures of large systems with a high degree of redundancy and do not present the total number of faults in the system, only those leading to a system failure.

In Figure 7.3 both the failures of internal system elements and the system failures are shown for a server site providing content on the web. Except for the operational faults, less than 11% of them result in system failures. There are few hardware faults, and all of them are tolerated, which may be interpreted as an indication of the state of the art. Note also the significant contribution of failures due to operation.

7.1.2 Approaches to dependable systems

There are two basic approaches to achieve a dependable system, namely *fault prevention* and *fault tolerance*. Fault prevention and fault tolerance may be regarded as barriers in the fault-error-failure path as illustrated in Figure 7.4.

Fault prevention

With *fault prevention*, the objective is a fault-free system.

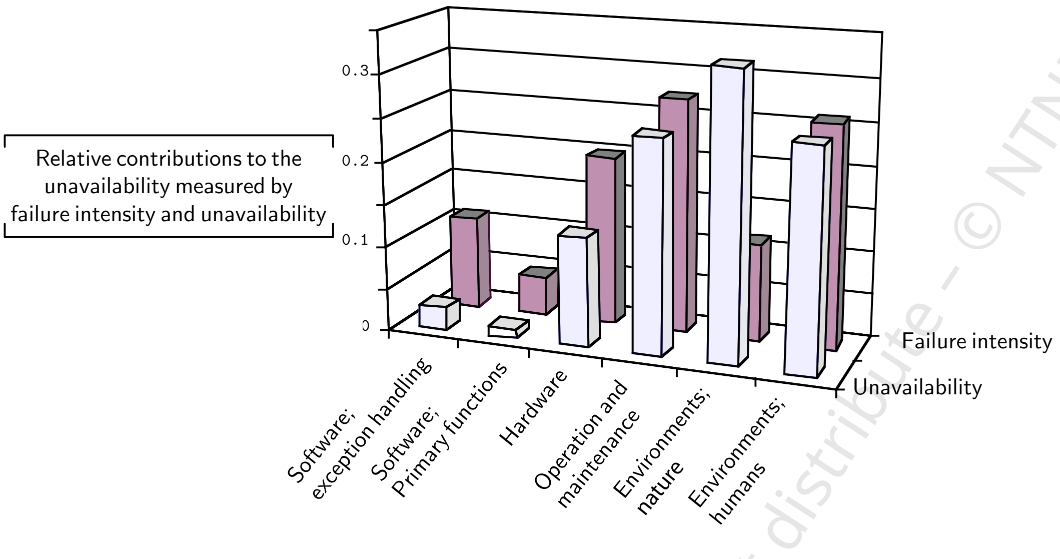


Figure 7.2: Sources of failure in the US public switched telephone network [41]

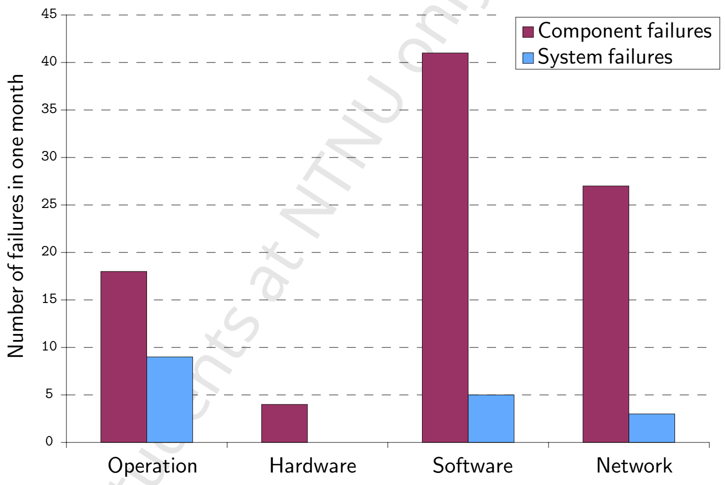


Figure 7.3: System element failures and resulting service failures for an Internet service cluster providing content [42]

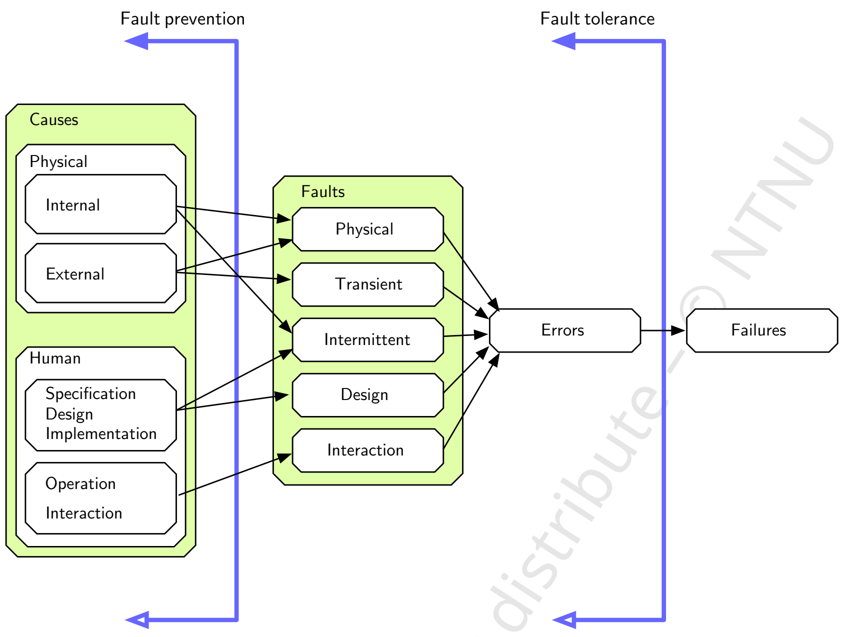


Figure 7.4: The design barriers of fault prevention and tolerance to improve dependability

With fault prevention no fault is introduced during design. The system is shielded against electromagnetic noise, dirt, dampness, etc. so no disturbance from the environment can cause faults or errors in the system. The rates of internal physical failures are reduced to a minimum by proven designs, careful manufacturing and derating. All operational actions are controlled by rigorous quality assurance procedures.

Fault tolerance

With *fault tolerance*, the objective is to deliver the specified service in spite of the existence and activation of faults in the system.

With fault tolerance an error is allowed to occur, but is prevented from causing a failure.

Fault-tolerance is realized by redundancy in

- *hardware*, e.g. spare or parallel components
- *software*, e.g. management or fault handling software
- *information*, e.g. error-detecting (typically CRC - Cyclic Redundancy Check) or error-correcting (typically FEC - Forward Error Control) codes
- *time*, e.g. operation control, comparison of parallel results, retransmission of erroneous packets

or any combination of the above improves the fault tolerance.

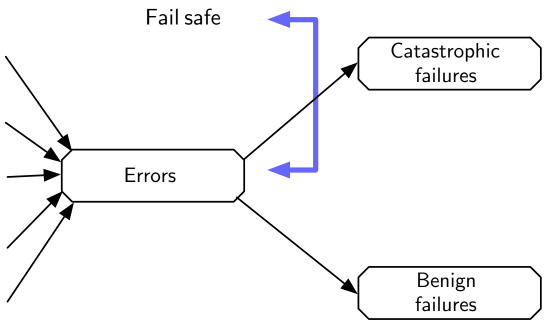


Figure 7.5: Fail-safe design approach

When safety is the most important dependability attribute of a system, a fail-safe design may be applied instead of a fault tolerant design. This approach is illustrated in Figure 7.5. The design seeks to make the benign failures the only possible. To enforce this, the detection of an error may immediately force the system into a state where it fails in a controlled and acceptable way. Hence, the reliability and availability of a safe system may be sacrificed for an improved safety.

Whether to use fault prevention or fault tolerance to meet the dependability requirements of a system, is not an either-or, but a balanced usage of both approaches. The objective is to develop a system with a minimum life-cycle cost. Hence, it is a matter of both development, production and equipment costs as well as operation and maintenance costs. General guidelines are not available, but some rough indication may be given.

- For systems with low dependability requirements, fault tolerance is always more costly in terms of development effort, additional hardware, and number of maintenance actions. (As a rule of thumb, the cost increases by at least a factor two). The dependability that can be achieved with fault prevention is limited by physically achievable component failure rates and repair times. For strict dependability requirements, only fault tolerance is capable of meeting them.
- Depending on the kind of system, it may be quite inexpensive to make the system tolerant to transient faults. An example is the application of error detection codes and retransmission on communication lines. In this case, the cost is some reduction in effective capacity and a small increase in complexity.
- With respect to design faults, fault prevention is the dominating approach. Rigid quality assurance, formal methods, specific design techniques, formal verification, extensive testing and automation of parts of the development process are among the fault prevention actions taken. Software fault tolerance has been a research issue for a number of years, but has limited practical application. A significant proportion of errors caused by design faults may however be tolerated by other means, see [9].

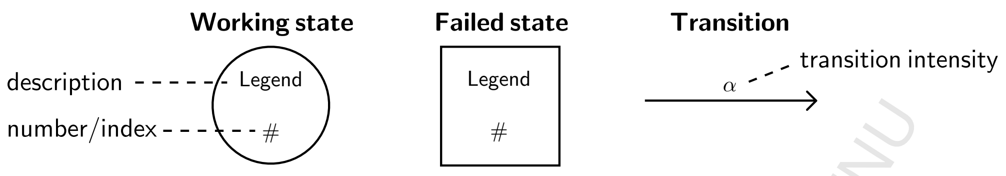


Figure 7.6: Notations for state transition diagrams

7.2 Markov models (state transition diagrams)

This section presents how to model the dependability of a system by a time continuous discrete space Markov process represented by a *state transition diagram*. Section 7.2.1 defines some notations. Sections 7.2.2 and 7.2.3 present example of dependability models and finally general results for Markov models are listed in Section 7.2.4.

7.2.1 Notations

The state space Ω is divided into:

Ω_W The set of *working* (up) states i.e. the set of states in which the system provides a set of services according the specification. These states are represented by *circles*, see Figure 7.6.

Ω_F The set of *failed* (down) states i.e. the set of states in which the system does not provide a set of services according the specification. These states are represented by *squares*, see Figure 7.6.

Ω_W and Ω_F form a partition of the state space, i.e. $\Omega = \Omega_W \cup \Omega_F$ and $\Omega_W \cap \Omega_F = \emptyset$.

The transition intensities between the states in Ω are denoted q_{ij} as described in Section 5.2.2. The steady state probabilities for the Markov model are denoted $p = \{p_1, \cdots, p_i, \cdots, p_n\}$. Methods for determining these probabilities are presented in Section 5.2.4.

Symbols used in state transition diagrams for dependability modeling are given in Figure 7.6.

7.2.2 Model of a non fault-tolerant system

A non fault-tolerant system is a system with no redundancy and only one failure mode. The system fails if a failure occurs. The simplest conceivable model of such a system is illustrated in Figure 7.7. It is a two state continuous time discrete space Markov model. Given that the system starts in the working state, it behaves as illustrated in Figure 1.13.

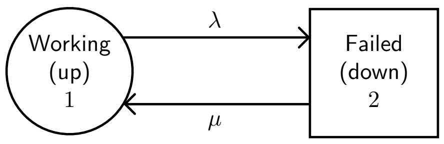


Figure 7.7: Two-state Markov model of the dependability of a system with no fault tolerance (no redundancy)

All times in the system are negatively exponentially distributed. Correspondingly, all failure and service restoration rates are constant. The system failure rate is denoted λ and the service restoration (repair) rate μ.

Availability

The notation p_W and $p_F = 1 - p_W$ is used for the probability of being in the up and down states, respectively. The asymptotic availability, see Section 1.2.2, is easily obtained from the equilibrium equation (in the long run, a stationary system fails as frequently as it is repaired, see Section 5.2.4).

$$\lambda p_W = \mu p_F \tag{7.1}$$

which, combined with the normalization condition ($p_W + p_F = 1$), gives

Asymptotic availability (two-state model)

$$A = p_W = \frac{1/\lambda}{1/\lambda + 1/\mu} = \frac{\mu}{\mu + \lambda} \tag{7.2}$$

This result corresponds to (1.3); since the up and down times are negatively exponentially distributed $MUT = \lambda^{-1}$ and $MDT = \mu^{-1}$. Besides, since there is only one up state, $MUT = MTFF$. Moreover, because of the memoryless property of the negative exponential distribution, see Section 5.1.3, and since there is only one up state, $MUT = MTFF = MTTF$.

To obtain the *instantaneous availability*, see Section 1.2.2, it is necessary to establish and solve a set of linear differential equations to capture the dynamic behavior of the system. Let $p_X(t)$ be the probability that the system is in state $X \in \{\text{Working}, \text{Failed}\} = \{W, F\}$ at time t. The probability that the system is in state X at time $t + \Delta t$ is the probability that it was in state X at time t and no failure or repair has occurred, or that it was in the other state and one failure or repair has occurred.

$$
\begin{aligned}
p_W(t+\Delta t) &= p_W(t)(1-\lambda\Delta t) + p_F(t)\mu\Delta t + o(\Delta t) \\
p_F(t+\Delta t) &= p_F(t)(1-\mu\Delta t) + p_W(t)\lambda\Delta t + o(\Delta t)
\end{aligned}
\tag{7.3}
$$

By definition,

$$\frac{\mathrm{d}}{\mathrm{d}t}p_X(t) = \lim_{\Delta t \to 0} \frac{p_X(t+\Delta t) - p_X(t)}{\Delta t} \tag{7.4}$$

Thus, the following system of first order linear differential equations is obtained

$$
\begin{cases}
\dfrac{\mathrm{d}}{\mathrm{d}t}p_W(t) = -\lambda p_W(t) + \mu p_F(t) & (7.5) \\[2ex]
\dfrac{\mathrm{d}}{\mathrm{d}t}p_F(t) = -\mu p_F(t) + \lambda p_W(t) & (7.6)
\end{cases}
$$

Assuming that the system is working at $t = 0$, i.e. $p_W(0) = 1$, and $p_F(0) = 0$ gives

Instantaneous availability (two-state model)

$$A(t) = p_W(t) = \frac{\lambda}{\mu+\lambda}e^{-(\lambda+\mu)t} + \frac{\mu}{\mu+\lambda} \tag{7.7}$$

Note that the time constant in the instantaneous availability is $(\lambda+\mu)^{-1}$, which for practical systems is dominated by μ, since $\mu \gg \lambda$. Hence, the information that the system was "as new" at $t = 0$ is lost in the same order of magnitude as the MDT.

Reliability function

The reliability function, see Section 1.2.2, is the probability that the system has been providing uninterrupted services up to t, i.e. that the first failure has not yet occurred at time t, $R(t) = P(T_{FF} > t)$. Since $T_{FF} \sim$ n.e.d.(λ) then

Reliability function (two-state model)

$$R(t) = P(T_{FF} > t) = e^{-\lambda t} \tag{7.8}$$

Since the model is Markovian (negatively exponentially distributed times, memoryless property, see Section 5.1.3) and has only one working state, the reliability function from an arbitrary point of time is the same as $R(t)$.

Reliability function from arbitrary point in time (two-state model)

$$\tilde{R}(t) = P(T_F > t) = e^{-\lambda t} = R(t) \tag{7.9}$$

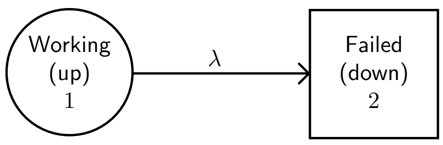


Figure 7.8: Dependability model with absorbing failed state

The reliability function may also be obtained from the Markov model. If the failed state is made absorbing (no transition out from this state, that is $\mu = 0$, or in other words, once the system enters this state it never leaves it) as illustrated in Figure 7.8 and given that the system starts in the working state $P_W(0) = 1$, the time to the first failure T_{FF} is the time the system spends in the working state. Then, (7.5) becomes

$$\frac{\mathrm{d}}{\mathrm{d}t} p_{W^*}(t) = -\lambda p_{W^*}(t) \tag{7.10}$$

and solves into $R(t) = P_{W^*}(t) = e^{-\lambda t}$.

7.2.3 A more complex system

In this section, a system with multiple components and fault causes is modeled. The model is constructed stepwise, starting with only permanent hardware failures and gradually extending it to include software and transient hardware failures and finally a standby (spare) component.

The system considered is a DNS (Domain Name System) server in the Internet. This system consists of two servers as illustrated in Figure 7.9. The servers are physically separated (e.g. in separate buildings) and connected to the network by two different access points (e.g. different routers). Hence, it can be assumed that the servers fail independently of each other. In addition, the operator has a spare server available that can be used as a standby server. This server is powered off and physically separated from the other two.

The configuration of the DNS servers ensures load sharing between the two servers. If the DNS users (end-users or DNS servers higher in the hierarchy) detect that a server is not responding, this server is classified as down and all the users are routed to the working server. The switch-over is ideal, i.e. it occurs immediately and never fails. If both servers are down, the system has failed and the DNS service is down. If available, the spare server may replace a failed server.

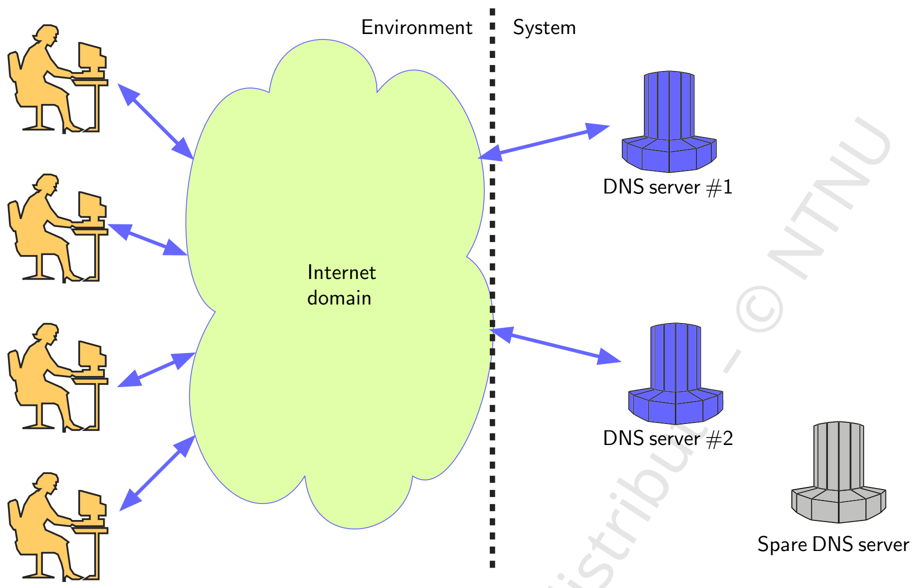


Figure 7.9: Internet DNS server system

The system is subject to permanent and transient hardware failures, and software failures. In addition, the servers are exposed to failures caused by environment. The DNS servers are maintained by one person who:

- repairs permanent hardware failures and failures caused by the environment (e.g. power outages) within $T_p \sim \text{n.e.d.}(\mu_p)$, and
- replaces a failed server by the spare server within $T_r \sim \text{n.e.d.}(\mu_r)$.

The servers have an embedded watchdog function which monitors the state of the machine. The watchdog detects when the server is not responding correctly and initiates reboot. In the case of a software or a transient hardware failure, rebooting repairs the server within T_t, which is assumed to be negatively exponentially distributed with parameter μ_t, $T_t \sim \text{n.e.d.}(\mu_t)$.

Permanent hardware failures

The first model only includes failures that require manual maintenance and repair, i.e. permanent hardware failures and failures caused by the environment, and there is no standby server. The failure rate of a server is λ_p.

The system is working as long as at least one server is working. The state X of the system is fully defined by the number of active servers $\Omega = \{0\text{p}, 1\text{p}, 2\text{p}\} = \{1, 2, 3\}$. A state transition diagram of the Markov dependability model of this system is shown in

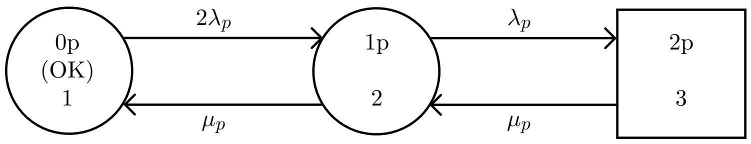


Figure 7.10: Markov dependability model of a DNS system with two servers subject to permanent hardware failures

Figure 7.10. In state 1 both servers are active and may fail, therefore the total failure rate is $2\lambda_p$. In state 2 one server is active can fail and the other one is under repair. In state 3 both servers have failed but since only one repairman is available only one is under repair and the transition rate is therefore μ_p (and not $2\mu_p$ as it would have been the case with more than one repairman).

The steady states probabilities $p_i = \lim_{t\to\infty} P(X(t) = i),\ i = 1, 2, 3$ are determined using the equilibrium equation (5.40) and the normalization condition $\sum_{i=1}^{3} p_i = 1$.

$$\begin{cases} 2\lambda_p p_1 = \mu_p p_2 \\ \lambda_p p_2 = \mu_p p_3 \\ p_1 + p_2 + p_3 = 1 \end{cases} \tag{7.11}$$

Solving this system gives the asymptotic unavailability

$$U = 1 - A = p_3 = \frac{2\lambda_p^2}{2\lambda_p^2 + 2\lambda_p\mu_p + \mu_p^2} \approx 2\left(\frac{\lambda_p}{\mu_p}\right)^2 \tag{7.12}$$

The approximation is valid under the assumption that $\mu_p \gg \lambda_p$. There is only one down state and the repair time is negatively exponentially distributed therefore $MDT = \mu_p^{-1}$. Hence, from (1.4)

$$MTBF = \frac{MDT}{U} \approx \frac{\mu_p}{2\lambda_p^2} \tag{7.13}$$

The reliability function is obtained by making state 3 absorbing as illustrated in Figure 7.11 and solving the system of differential equations.

Software and transient hardware failures

Software failure have in most systems a significant impact on the dependability. In order to increase the realism of the model, both the software and the transient hardware

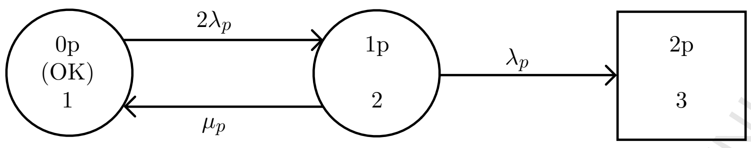


Figure 7.11: Markov dependability model of a DNS system with two servers subject to permanent hardware failures and with an absorbing failed state

failures are now included. These failures are referred to as temporary failures and occur according to a Poisson process of intensity λ_t. The restoration after a temporary failure does not require (manual) maintenance and repair, it is assumed that the automatic reboot fix the problem. It is also assumed that a server with a permanent hardware failure cannot fail due to temporary failures. The opposite it however possible, a server with a temporary failure can fail due to a permanent hardware failure. When it is repaired within time T_p both permanent and temporary failures are removed. The spare server is still not included in the model.

The state space is now two dimensional. In addition to the number of servers failed, it is necessary to distinguish between the two types of failures.

$$\Omega = \{\{i\text{p}, j\text{t}\} \mid i \in \{0,1,2\}, j \in \{0,1,2\}, i + j \leqslant 2\} \tag{7.14}$$

A state transition diagram of the Markov dependability model of this system where each state is assigned a number to simplify the notation, $\Omega = \{1,2,3,4,5,6\}$, is shown in Figure 7.12. This model is an extension of the model in Figure 7.10.

Note that the servers are restarted independently of each other. Therefore when two temporary failures have occurred (state 6), the total restart rate is $2\mu_t$ (transition from state 6 to state 4).

In state 4, a temporary failure has occurred on one of the servers and the other server is working. Then, either

- the working server fails due to a permanent hardware failure and the system changes to state 5,
- the failed server fails due to a permanent hardware failure and the system changes to state 2,
- the restart is successful and the system is back in the initial state 1, or
- a temporary failure occurs on the working server and the system changes to state 6.

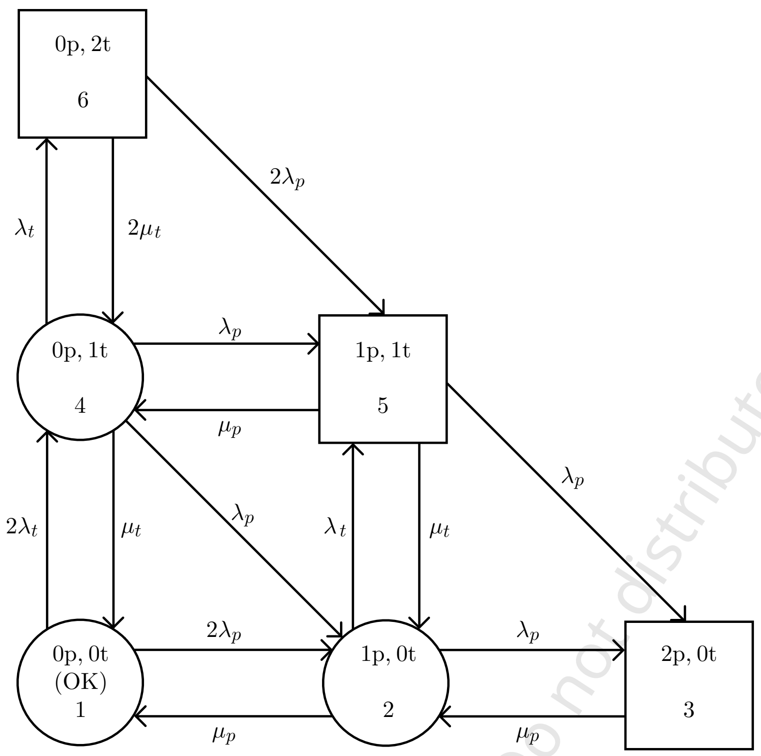


Figure 7.12: Markov dependability model of a DNS system with two servers subject to software failures and permanent and transient hardware failures

The transition intensity matrix for the model, see Section 5.2.4, is

$$\underline{Q} = \begin{bmatrix} -2\lambda_p - 2\lambda_t & 2\lambda_p & 0 & 2\lambda_t & 0 & 0 \\ \mu_p & -\mu_p - \lambda_p - \lambda_t & \lambda_p & 0 & \lambda_t & 0 \\ 0 & \mu_p & -\mu_p & 0 & 0 & 0 \\ \mu_t & \lambda_p & 0 & -\mu_t - 2\lambda_p - \lambda_t & \lambda_p & \lambda_t \\ 0 & \mu_t & \lambda_p & \mu_p & -\mu_t - \lambda_p - \mu_p & 0 \\ 0 & 0 & 0 & 2\mu_t & 2\lambda_p & -2\lambda_p - 2\mu_t \end{bmatrix} \tag{7.15}$$

The steady state probabilities $\underline{p} = \{p_1, \cdots, p_6\}$ are obtained by solving $\underline{p} \cdot \underline{Q} = \underline{0}$, where one of the columns is substituted by the normalization condition $\sum_{i=1}^{6} p_i = 1$.

The asymptotic unavailability is equal to the sum of the steady state probabilities of the failed states.

$$U = p_3 + p_5 + p_6 \tag{7.16}$$

Large sets of state equations are typically solved numerically because a symbolic solution is too demanding. However, for the simple example presented here, a symbolic solution may be obtained. In addition, due to symmetries in the model, it has a rather simple closed-form solution.

$$U = \frac{2\lambda_p^4 + \lambda_t^2\mu_p^2 + 4\lambda_p^3(\lambda_t+\mu_t) + 2\lambda_p\lambda_t\mu_p(\lambda_t+\mu_t) + 2\lambda_p^2(\lambda_t^2+\mu_t^2+\lambda_t(\mu_p+2\mu_t))}{(\lambda_p+\lambda_t+\mu_t)^2(2\lambda_p^2+\mu_p^2+2\lambda_p\mu_p)} \tag{7.17}$$

Typically, the repair rates are much larger (e.g. a factor of 10^3) than the failure rates. Hence, the unavailability can be approximated by

$$U \approx 2\left(\frac{\lambda_p}{\mu_p}\right)^2 + \left(\frac{\lambda_t}{\mu_t}\right)^2 + 2\frac{\lambda_p\lambda_t}{\mu_p\mu_t},\ \mu_t,\mu_p \gg \lambda_t,\lambda_p \tag{7.18}$$

The first term is recognized from (7.12) as the unavailability due to permanent failures, the second term is the unavailability due to temporary failures, and the last term is due to the combination of both types of failures.

The calculations of *MDT* and *MTBF* are presented in Section 7.2.4.

Lukewarm standby server

To improve the dependability of the system shown in Figure 7.9, a standby server is included. This server is put into operation to replace a server that has failed due to a permanent hardware failure. The contribution of permanent hardware failures to the unavailability of the system is reduced this way, and so is the down time.

An unpowered spare component is called a *cold standby*. It is common to (optimistically) assume that such a standby component never fails. In this example, a more realistic approach is taken by assuming that the standby component (the spare server) fails according to a Poisson process of intensity $\lambda_r > 0$ and may fail even when it is inactive. Such a spare component is called a *lukewarm standby*.

The possible states of the standby server are:

0 – the server is operational.

x – the server has been exposed to a permanent fault which will manifest itself as a failure when the server is put into operation (*dormant fault*). This state may occur only when both primary servers are working.

1 – there is a known permanent failure in the server.

Adding the standby server adds one dimension to the state space.

$$\Omega = \{\{i\text{p}, j\text{t}, k\text{r}\} \mid i \in \{0,1,2\}, j \in \{0,1,2\}, i+j \leqslant 2, k \in \{0,x,1\}, k=x \implies i=0\} \tag{7.19}$$

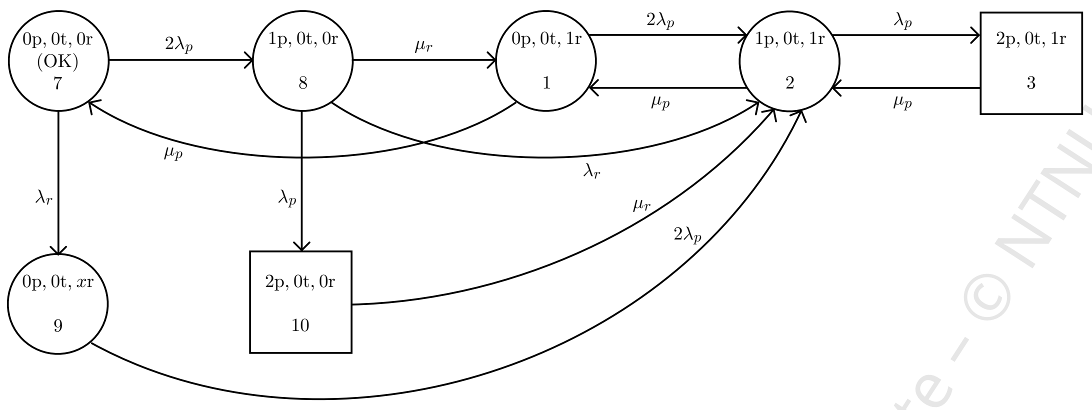


Figure 7.13: Markov dependability model of a DNS system with two primary servers and a lukewarm standby server subject to permanent hardware failures

The time to replace a failed primary server by the lukewarm standby is denoted T_r and is assumed negatively exponentially distributed with parameter μ_r.

In Figure 7.13 the model shown in Figure 7.10 is extended. Temporary failures are not considered. The new states are enumerated 7, 8, 9 and 10. The initial state (everything is working) is now state 7.

The transition from state 7 to state 9 models the occurrence of a dormant fault in the lukewarm standby server. In state 9 the failure of one of the primary servers brings the system to state 2 where both one primary server and the lukewarm standby server have failed. State 8 represents the system state where a primary server has failed and the standby server is not yet put into operation. The transition to state 10 models the failure of the remaining active server. The transitions from state 8 to state 1 and from state 10 to state 2 model the replacement of the primary server that has failed by the standby server. Finally, the transition from state 1 to state 7 models the repair of the standby server.

Again, the asymptotic unavailability is obtained from set of state equations obtained by applying the equilibrium equation. Assuming that $\mu_r, \mu_p \gg \lambda_r, \lambda_p$ then

$$U \approx \frac{\lambda_r}{2\lambda_p + \lambda_r} \cdot 2\left(\frac{\lambda_p}{\mu_p}\right)^2 + \frac{\lambda_p}{2\lambda_p + \lambda_r} \cdot 4\left(\frac{\lambda_p}{\mu_r}\right)^2 + 4\left(\frac{\lambda_p}{\mu_p}\right)^3, \ \mu_r, \mu_p \gg \lambda_r, \lambda_p \qquad (7.20)$$

In (7.20)

- the last term is of higher order than the two first terms. This means that this term only contributes when the standby server failure rate λ_r is very small ($\lambda_r \to 0$) and the time μ_r^{-1} to replace a failed primary server by the standby server is very short ($\mu_r \to \infty$). An ideal standby (cold standby) never fails and then $U \approx 4(\frac{\lambda_p}{\mu_p})^3$.

- the first term dominates if the time μ_r^{-1} to replace a failed primary server by the standby server is very short ($\mu_r \to \infty$) and the failure rates of primary and standby servers are in the same order of magnitude, $\lambda_p \sim \lambda_r$.
- the second term dominates when the standby server failure rate is very small ($\lambda_r \to 0$) and the repair time and the time to replace a failed primary server by the standby server in the same order of magnitude, $\mu_p \sim \mu_r$.

In Figure 7.14 the lukewarm standby server is included and both permanent hardware failures and temporary failures are taken into account. This new model extends the models presented in Figures 7.12 and 7.13. The transitions from state 7 to state 13, from state 13 to state 15, and from state 8 to state 14 represent the occurrence of a temporary failure in one of a primary servers while the standby server is operational. The transitions from state 9 to state 11 and from state 11 to state 12 represent the occurrence of a temporary failure in a primary server while there is a latent error in the standby server.

7.2.4 General results

In this section we introduce some general dependability results that can be obtained from Markov state transition diagrams.

Asymptotic availability

The asymptotic (steady state) availability is the probability that a system provides a set of services at a given instant of time when the system is in stationary state.

Asymptotic availability in Markov models

$$A = \sum_{i \in \Omega_W} p_i \tag{7.21}$$

Similarly, the asymptotic unavailability is

Asymptotic unavailability in Markov models

$$U = \sum_{i \in \Omega_F} p_i \tag{7.22}$$

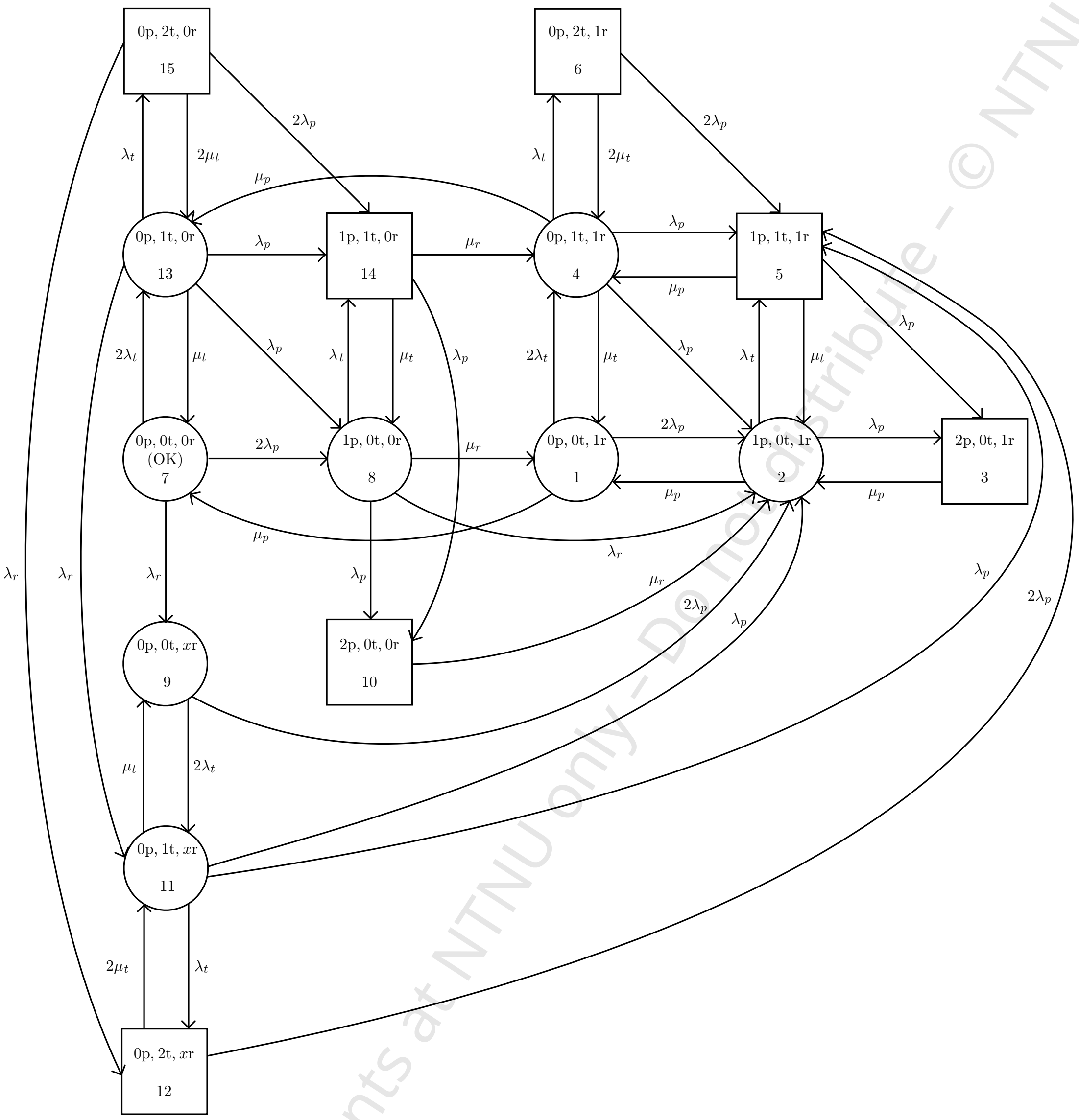


Figure 7.14: Markov dependability model of a DNS system with two primary servers and a lukewarm standby server subject to software failures and permanent and transient hardware failures

System times

According to the equilibrium equation (5.40), observing the number of transitions in a stationary system over a long period of time from one subset of states to the complementary subset is equal to (or differs at most by one) the number of transitions observed in the opposite direction. By partitioning the state space Ω into the two subsets Ω_W and Ω_F, the number of failures in the long run are equal to the number of repairs in the system. Hence, the system failure intensity, Λ, is obtained either from the transitions from the working to the failed set, or vice-versa.

System failure intensity in Markov models

$$\Lambda = \sum_{i\in\Omega_W}\sum_{j\in\Omega_F} q_{ij}p_i = \sum_{i\in\Omega_W}\sum_{j\in\Omega_F} q_{ji}p_j \tag{7.23}$$

Applying Korolyuk's theorem (3.32) gives

$$MTBF = \frac{1}{\Lambda} \tag{7.24}$$

The mean up and down times are obtained using Little's formula, see Section 3.4

$$MUT = A \cdot MTBF \tag{7.25}$$

$$MDT = U \cdot MTBF \tag{7.26}$$

Reliability function

The reliability function and $MTFF$ and $MTTF$, see Section 1.2.2, are determined by a slightly more complicated procedure. The approach described in this section consists in modifying the model.

First, make all the failed states absorbing by letting all transitions out of Ω_F be zero.

$$q_{ij} \to 0, \forall i \in \Omega_F \tag{7.27}$$

Now, without any implication on the system behavior in the working states, all the failed states can be merged into one because the system never leaves any of the failed (absorbing) states when first entered. For instance, the model shown in Figure 7.15 is the modification of the model presented in Figure 7.12 .

Section 5.2.1 describes how to determine the transient state probabilities for a Markov model. In the following, the transient state probabilities, $p_i(t)$, $\forall i \in \Omega$, are assumed to be known in a system where $q_{ij} \to 0,\ \forall i \in \Omega_F$.

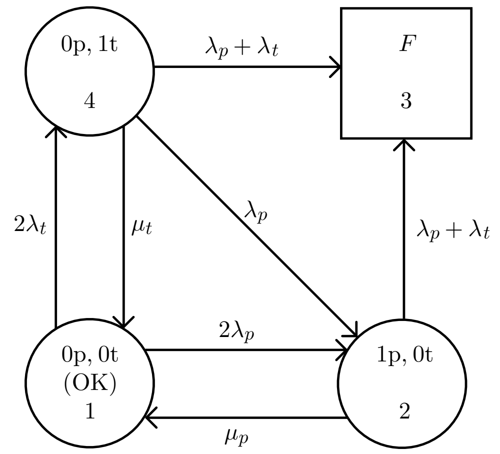


Figure 7.15: Markov dependability model of a DNS system with two servers subject to software failures and permanent and transient hardware failures with absorbing failed states; see Figure 7.12 for the original model

The time to first failure T_{FF} is the time from the system is new, or as new, until the first failure is observed. The mean time to first failure is

$$MTFF = \mathrm{E}(T_{FF}) = \int_0^\infty P(T_{FF} > t)\,\mathrm{d}t = \int_0^\infty R(t)\,\mathrm{d}t \tag{7.28}$$

where $R(t)$ is the reliability function given that the system is new at time $t = 0$.

When all failed states are absorbing, the probability that the system is working at time t, i.e. $R(t) = P(T_{FF} > t)$, is equal to the probability that the system is in one of the working states at time t.

Reliability function in Markov model

If the system is new, or as new, at time $t = 0$

$$R(t) = \sum_{i \in \Omega_W} p_i(t),\ p_i(0) = \begin{cases} 1 & \text{if } i = 1 \text{ (OK)} \\ 0 & \text{otherwise, } i \in \Omega - \{1\} \end{cases} \tag{7.29}$$

Now assume that the system is working and stationary at $t = 0$. The time to failure T_F and the corresponding mean time to failure is

$$MTTF = \mathrm{E}(T_F) = \int_0^\infty P(T_F > t)\,\mathrm{d}t = \int_0^\infty \tilde{R}(t)\,\mathrm{d}t \tag{7.30}$$

where $\tilde{R}(t)$ is the reliability function given that the system is stationary and in a working state at time $t = 0$.

When all failed states are absorbing, the probability that the system is working at time t, i.e. $R(t) = P(T_F > t)$, is equal to the probability that the system is in one of the working states at time t.

Reliability function in Markov model

If the system is working and stationary at time $t = 0$

$$\tilde{R}(t) = \sum_{i \in \Omega_W} p_i(t),\ p_i(0) = \begin{cases} p_{W_i} & \text{if } i \in \Omega_W \\ 0 & \text{otherwise, } i \in \Omega_F \end{cases} \tag{7.31}$$

where p_{W_i} is the conditional steady state probability that the system is in state i given that the system is working. The p_{W_i}s are determined from the steady state probabilities calculated for the unmodified system.

$$p_{W_i} = P(I = i \mid I \in \Omega_W) = \frac{P(I = i)}{P(I \in \Omega_W)} = \frac{p_i}{\sum_{j \in \Omega_W} p_j},\ i \in \Omega_W \tag{7.32}$$

Simplified procedure to determine $MTFF$ and $MTTF$

It is possible to find $MTFF$ and $MTTF$ without solving the system of differential equations for $p_i(t),\ \forall i \in \Omega_W$. The approach described in this section consists in modifying the state transition diagram. For another approach, see for instance [43].

Procedure to determine $MTFF$

1. Make all the failed states absorbing and merge them into a unique failed state denoted F.
$$q^*_{iF} = \sum_{j \in \Omega_F} q_{ij},\ i \in \Omega_W \tag{7.33}$$
The transition intensities between the working states remain unchanged.
$$q^*_{ij} = q_{ij},\ (i,j) \in \Omega_W \tag{7.34}$$
This first step is illustrated in Figure 7.16 and is identical to the modification of model applied to obtain the reliability function in the previous section.
2. Add a transition from state F to the initial state with intensity $q^*_{F,\text{OK}} = q_0$. See Figure 7.17 for an example.
3. Determine the steady state probability p^*_F that the system is in state F.

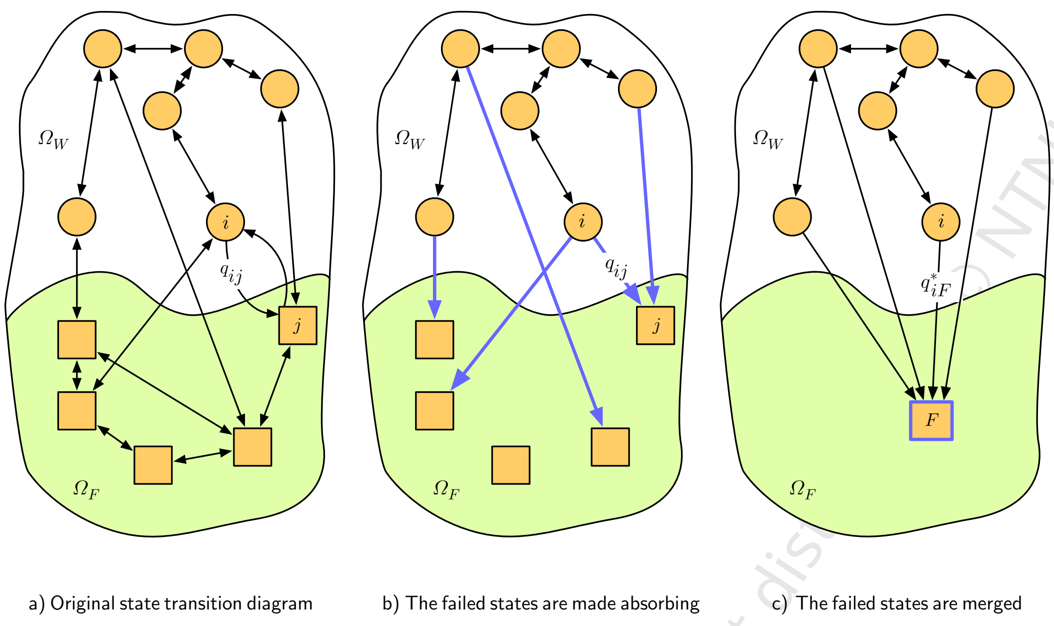

Ω_W
i
q_{ij}
j
Ω_F
q^*_{iF}
F
a) Original state transition diagram
b) The failed states are made absorbing
c) The failed states are merged

Figure 7.16: Modification of the state transition diagram

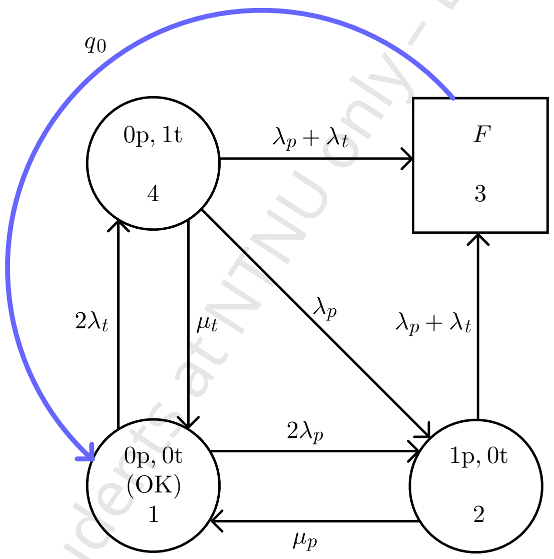


Figure 7.17: Modification of the state transition diagram shown in Figure 7.12 to obtain $MTFF$.

4. Calculate $MTFF$ as
$$MTFF = \frac{1 - p_F^*}{p_F^* q_0} \tag{7.35}$$
The modification of the model does not change the model of the system behavior in the working state space. Thus, the mean time to first failure in the original and the modified models are identical, $MTFF = MTFF^*$. Furthermore, since a repair in the modified model always leads back to the initial state, the mean time to first failure is also equal to the mean up time.
$$MTFF = MTFF^* = MUT^* \tag{7.36}$$
In addition, since F is the only failed state, from (7.22) $U^* = p_F^*$ and from (7.24) $\Lambda^* = p_F^* q_0$. Therefore from (7.25)
$$MUT^* = (1 - U^*)MTBF^* = \frac{(1 - p_F^*)}{p_F^* q_0} \tag{7.37}$$
Combining (7.36) and (7.37), we obtain (7.35). p_F^* depends on q_0 and q_0 cancels in (7.35).

Procedure to determine $MTTF$

1. Make all the failed states absorbing and merge them into a unique failed state denoted F.
$$q_{iF}^\dagger = \sum_{j \in \Omega_F} q_{ij},\ i \in \Omega_W \tag{7.38}$$
The transition intensities between the working states remain unchanged.
$$q_{ij}^\dagger = q_{ij},\ (i,j) \in \Omega_W \tag{7.39}$$
This step is identical to the first step of the procedure to determine $MTFF$ described above.
2. Add a transition from state F to every state $i \in \Omega_W$ (working state) with intensity $q_{F,i}^\dagger = q_0 p_{W_i}$. That is the probability that the system is restored to a given working state is given by the steady state probabilities of the original model. See Figure 7.18 for an example. This way $MTTF^\dagger = MUT^\dagger$.
3. Determine the steady state probability $p_F^\dagger$ that the system is in state F.
4. Calculate $MTTF$ as
$$MTTF = \frac{1 - p_F^\dagger}{p_F^\dagger q_0} \tag{7.40}$$
This relation is obtained from a reasoning similar to the one used to obtain (7.35) given that now $MTTF = MTTF^\dagger = MUT^\dagger$.

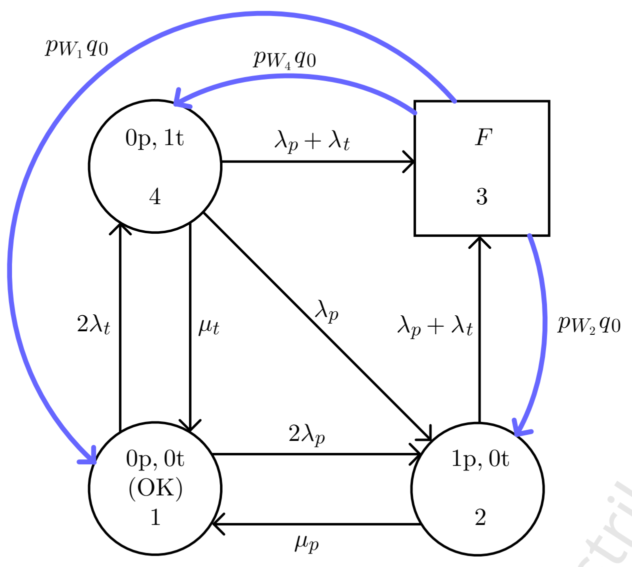
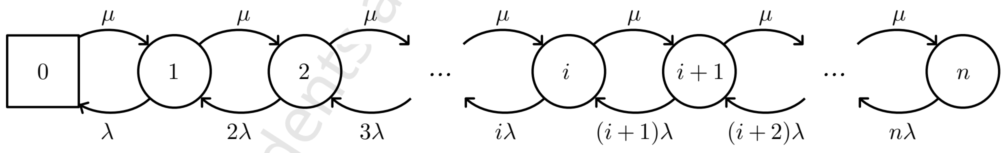


Figure 7.18: Modification of the state transition diagram shown in Figure 7.12 to obtain $MTTF$.

Example 7.1 Model of a fault-tolerant system with a regular structure
Consider a system consisting of n identical components with failure rate λ. The components fail independently of each other. The repair rate of failed components is μ. If there are more than one failure in the system at the same time, these are handled one by one. This is the case with only one repairman for the system, or with only one physical disk where all the reboot software and configuration data are stored.
If the number of working components is chosen as the system state variable, then the Erlang's loss model is recognized where the repair rate μ corresponds to the arrival rate λ and the failure rate $\lambda_i = i\lambda$ corresponds to the service rate $\mu_i = i\mu$ in the Erlang's loss model presented in Section 6.3.1. The model of the system given in the figure below.



The Erlang's loss formula $E_n(\mu/\lambda)$ gives the probability that all the components are working.
If at least one of the n components must be working for the system to be working, the asymptotic unavailability of the system is given by

$$U = p_0 = \left(\sum_{\nu=0}^{n} \frac{\mu}{\nu! \lambda} \right)^{-1}$$

7.3 Structural models

A system may consist of a number of subsystems that interwork to deliver the services provided by the system. In a fault-tolerant system, some of these subsystems may fail without causing system failure.

In Section 7.2 the structure of the system is not explicitly taken into consideration. Markov models allow a detailed modeling of the system behavior and complicated correlation and dependencies may be included, but, for large systems, the state transition diagrams are graphically impractical and the models become computationally very demanding or even in some cases unsolvable.

This section presents an alternative approach that enables the modeling and evaluation of the dependability of large systems based on the structure of the system. The graphical model representation can handle quite large systems without becoming unreadable, and the computations become rather simple, in particular for the *parallel-series structures*. However, this comes at the cost of restrictive assumptions made about the system and its behavior, and system dynamics and dependencies are to a great extent ignored in structural models.

Note that assumptions about the system and its behavior are only required for quantitative analysis of the dependability of the system, but are not required for qualitative analysis, see Section 7.3.4.

In this section, two structural modeling methods are presented, namely reliability block diagrams in Section 7.3.1 and fault-trees in Section 7.3.2. Section 7.3.3 gives the relation between the two methods. Finally, Section 7.3.4 presents techniques for qualitative dependability analysis based on structural models.

In the following, $X_i(t) \in \{\text{Working}, \text{Failed}\} = \{W, F\}$ represents the state of a system component i at time t. The state of a system consisting of n components is given by the state vector $\underline{X}(t) = \{X_1(t), \ldots, X_n(t)\}$. $\underline{X}(t) \in \Omega = \{\Omega_W, \Omega_F\}$ where Ω_W and Ω_F denote the set of working states and the set of failed states, respectively. To simplify the notation the reference to time is omitted unless required for the understanding.

7.3.1 Reliability block diagrams

Reliability block diagrams are used to model, depict and analyze the structure of a system with respect to dependability. The strength of reliability block diagrams is their ability to reflect the design of the system and serve as a constructive tool.

As mentioned in the introduction to this section, structural models simplify the modeling of large and complex systems compared to Markov models, but this comes at the cost of restrictive assumptions made about the system and its behavior. The assumptions

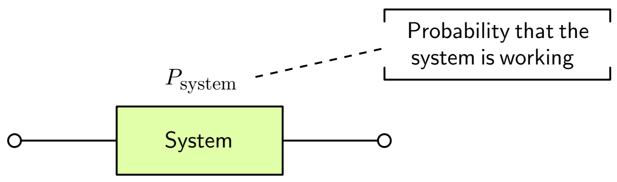


Figure 7.19: A reliability block

made when using reliability block diagrams to compute dependability attributes are listed below.

1. If the system is made up by more than one subsystem, each system fails independently of all other subsystems, i.e. independently of the state of the system.
2. If a subsystem has failed, its service is restored independently of the state of all other subsystems. (This assumption requires for instance a dedicated repairman and test equipment for each subsystem.)
3. The system behaves as intended. For instance, a fault-tolerant system handles faults as long as there are remaining (spare) resources, service restoration actions are successful, and failures (errors) do not propagate from one subsystem to others.

Note that if any of these assumptions deviates from the true behavior of the system, the analysis loses its precision.

The basic entity of a reliability block diagram is the *reliability block*, illustrated in Figure 7.19. This block represents a *system*, i.e. an entire system providing services, its *subsystems* and so on down to the smallest system component considered in the model ("atomic system"). A given block must be included only once in a model. A block may be seen as a relay that connects two end-points (represented by small circles) with the probability P_{system}. When connection is provided between two end-points the system works, otherwise the system is failed.

Reliability block diagrams can be used to analyze both the availability and the reliability of a system. Depending on the analysis carried out, "works" is interpreted differently and P_{system} represents different properties of the system.

- *Availability analysis*, which is carried out to find A[2]. In this case, "works" should be interpreted as "provides service at a given instant of time", i.e. $P_{\text{system}} \equiv A(t) = P(X(t) \in \Omega_W)$ to determine the instantaneous availability and $P_{\text{system}} \equiv A = P(X \in \Omega_W)$ to determine the asymptotic availability.

[2]Availability analysis may also be used to find MUT, MDT and $MTBF$. This is not treated here, see for instance [9] for an introduction.

- *Reliability analysis,* which is carried out to find $R(t)$ and $MTFF$. In this case, "works" should be interpreted as "the system has provided uninterrupted service in the period $[0, t]$", i.e. $P_{\text{system}} \equiv R(t) = P(X(\tau) \in \Omega_W, \forall \tau \in [0, t])$. The reliability of a system cannot be analyzed with a reliability block diagram when some of the blocks representing subsystems are allowed to be repaired. This interpretation is global in a diagram and requires that no subsystem can be repaired.

Structure function

The state of a system component can be regarded as a Boolean[3] variable X_i defined as

$$\begin{aligned} X_i = W \Leftrightarrow \mathsf{X}_i = \text{True} \\ X_i = F \Leftrightarrow \mathsf{X}_i = \text{False} \end{aligned} \tag{7.41}$$

The corresponding boolean system state vector is $\underline{\mathsf{X}} = \{\mathsf{X}_1, \ldots, \mathsf{X}_n\}$.

The structure function $\Phi(\underline{\mathsf{X}})$ of a system is a Boolean function defined as

Structure function

$$\Phi(\underline{\mathsf{X}}) = \begin{cases} \text{True} & \text{if } \underline{X} \in \Omega_W \\ \text{False} & \text{if } \underline{X} \in \Omega_F \end{cases} \tag{7.42}$$

In other words, $\Phi(\underline{\mathsf{X}})$ is true if the system works, false otherwise.

Note that the structure function maintains the identity of all the system elements. Hence, it cannot be reduced based on symmetries in the system, functional identity of subsystems (identical subsystems), or statistical independence between subsystems. As opposed to quantitative dependability metrics calculated using reliability block diagrams, the structure function itself does not depend on

- the system element behavior (e.g. distributions of time to failure, series correlations in the failure process, distribution of repair times), or
- the statistical inter-dependence between the system elements (e.g. limited repair resources, repair strategies, common faults).

In dependability simulations, the structure function of a system can be used to determine whether the system is working or not, without any restrictions on the simulation model except that the structure of the system is retained.

[3]George Boole (1815-1864) is the inventor of Boolean algebra. A Boolean algebra is a set supplied with two binary operations $\wedge$ (logical AND), $\vee$ (logical OR), a unary operation $\neg$ (logical NOT) and two distinct elements 0 (zero/False) and 1 (one/True). In this book, the electronic engineering notation is adopted; $\cdot$, $+$ and $^{-}$ are used for logical AND, logical OR and logical NOT, respectively.

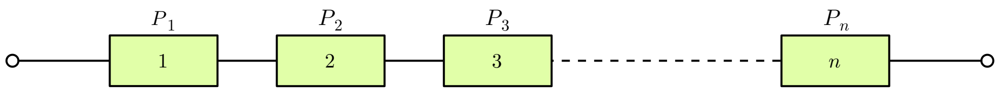
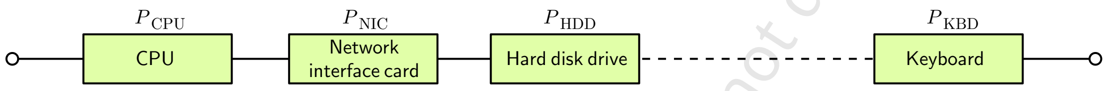


Figure 7.20: Reliability block diagram of a series system

Series system

A series system is a system where all the subsystems forming the system must work for the system to work (non fault-tolerant system). A series system is depicted as in Figure 7.20.

Example 7.2 Model of a PC
Consider a PC consisting of a motherboard, a cache, a memory, a disc, an internal bus, a network card, a power supply, a monitor, a keyboard, etc. All the elements must be working for the PC to be working. Hence, the PC has a series structure.
The reliability block diagram for this system is shown in the figure below.



Formally, a system S composed of n subsystems $S = \{S_1, S_2, \ldots, S_n\}$ has a series structure if all n subsystems are required to be working for the system to be working.

Structure function of a series system

$$\Phi_{\text{series}}(\underline{\mathsf{X}}) = \mathsf{X}_1 \cdot \mathsf{X}_2 \cdot \ldots \cdot \mathsf{X}_n = \prod_{i=1}^{n} \mathsf{X}_i \quad (7.43)$$

Since all the subsystems are assumed to be statistically mutually independent, the asymptotic availability is

Asymptotic availability of a series system

$$A_{\text{series}} = \prod_{i=1}^{n} A_i \quad (7.44)$$

and the reliability function is

Reliability function of a series system

$$R_{\text{series}}(t) = \prod_{i=1}^{n} R_i(t) \quad (7.45)$$

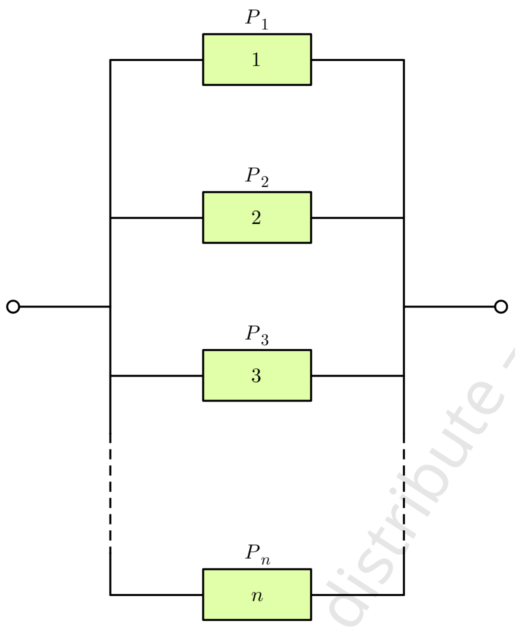


Figure 7.21: Reliability block diagram of a parallel system

If the diagram is used for reliability analysis and the time to first failure of every subsystem is negatively exponentially distributed with parameter λ_i, from (7.45)

$$R_{\text{series}}(t) = \prod_{i=1}^{n} e^{-\lambda_i t} = e^{-\lambda_{\text{series}} t} \tag{7.46}$$

where $\lambda_{\text{series}} = \sum_{i=1}^{n} \lambda_i$ is the failure rate for the entire system, see Section 5.1.4, and the mean time to first failure is

MTFF of a series system (with constant failure rates λ_i)

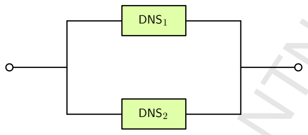
$$MTFF_{\text{series}} = \frac{1}{\lambda_{\text{series}}} \tag{7.47}$$

Parallel systems

A parallel system is a system where at least one the subsystems forming the system must work for the system to work (fault-tolerant system). Systems with a parallel structure have additional (redundant) system components to increase their dependability. Different forms of redundancy are discussed in [9]. A parallel system is depicted as in Figure 7.21.

Example 7.3 DNS server
The DNS server system presented in Figure 7.9 with only permanent hardware failures and without a standby server cannot be modeled by a reliability block diagram because the states of the two servers are not independent (because there is only one repairman).

If the two DNS servers can be repaired independently of each other, e.g. by introduction of two separate management organizations, then the system can be modeled by a reliability block diagram as shown in the figure opposite.



Formally, a system S composed of n subsystems $S = \{S_1, S_{2,} \cdots S_n\}$ has a parallel structure if at least one of the n subsystems must be working for the system to be working.

Structure function of a parallel system

$$\Phi_{\text{parallel}}(\underline{\mathsf{X}}) = \mathsf{X}_1 + \mathsf{X}_2 + \ldots + \mathsf{X}_n = \sum_{i=1}^{n} \mathsf{X}_i \tag{7.48}$$

To obtain the asymptotic availability, A_{parallel}, the easiest approach is to first determine the unavailability U_{parallel}, i.e. the probability that the system is not working. The system is not working when all its subsystems have failed. Since the subsystems are assumed to be statistically mutually independent, this is $U_{\text{parallel}} = \prod_{i=1}^{n} U_i$.

The unavailability can also be obtained by reorganizing the structure function (7.48) using De Morgan's law[4], which yields

$$\overline{\Phi_{\text{parallel}}(\underline{\mathsf{X}})} = \overline{\mathsf{X}}_1 \cdot \overline{\mathsf{X}}_2 \cdots \overline{\mathsf{X}}_n = \prod_{i=1}^{n} \overline{\mathsf{X}}_i \tag{7.49}$$

The unavailability of the system is $U_{\text{parallel}} = P(\overline{\Phi_{\text{parallel}}(\underline{\mathsf{X}})}) = \prod_{i=1}^{n} P(\overline{\mathsf{X}}_i) = \prod_{i=1}^{n} U_i$.

Now, substituting $U = 1 - A$ gives

Asymptotic availability of a parallel system

$$A_{\text{parallel}} = 1 - \prod_{i=1}^{n}(1 - A_i) \tag{7.50}$$

Likewise, the reliability function is

Reliability function of a parallel system

$$R_{\text{parallel}}(t) = 1 - \prod_{i=1}^{n}(1 - R_i(t)) \tag{7.51}$$

[4]Augustus De Morgan (1806-1871): $\overline{A + B} = \bar{A} \cdot \bar{B}$ and it dual form $\bar{A} + \bar{B} = \overline{A \cdot B}$

If all the subsystems in a parallel system are identical and have the same constant failure rate, $\lambda_i = \lambda$, i.e. the reliability function of each subsystem is $R_i(t) = e^{-\lambda t}, \forall i$, the reliability function for the entire system is

$$R_{\text{parallel}}(t) = 1 - \prod_{i=1}^{n}(1 - e^{-\lambda t}) = 1 - (1 - e^{-\lambda t})^n \tag{7.52}$$

The system time to first failure does not follow a negative exponential distribution. Hence, the system failure rate is not constant and should not be interpreted as the inverse of the $MTFF_{\text{parallel}}$ as it was the case in (7.47). Deriving the $MTFF_{\text{parallel}}$ directly from (1.10) and (7.51) is a bit demanding. However, there is a straightforward derivation from a physical interpretation of the behavior of the system. When there are i subsystems working, each having a constant failure rate of λ, the expected time to the failure of any working subsystem is $(i\lambda)^{-1}$. The system starts with n subsystems in parallel. These fail one by one. It then follows that

MTFF of a parallel system (with equal failure rates λ)

$$MTFF_{\text{parallel}} = \frac{1}{\lambda}\sum_{i=1}^{n}\frac{1}{i} \tag{7.53}$$

The positive gain achieved by adding identical system components in parallel decreases as the number of components increases. The same effect is illustrated for $R_{\text{parallel}}(t)$ in Figure 7.22. Note that the more redundant blocks the longer the period (from $t = 0$) the system reliability function is close to 1.

k-out-of-n system

A k-out-of-n system is a system where at least k of the n subsystems forming the system must work for the system to work, $1 \leqslant k \leqslant n$. A k-out-of-n system is depicted as in Figure 7.23. When $k = 1$, it is a parallel system, and when $k = n$, it is a series system.

Formally, a system S composed of n subsystems $S = \{S_1, S_{2,} \cdots S_n\}$ has a k-out-of-n structure if k of the n subsystems are required to be working for the system to be working.

The structure function of a k-out-of-n system includes all the possible combinations of k subsystems out of n.

$$\begin{aligned}\Phi_{k-\text{out}-\text{of}-n}(\underline{\mathsf{X}}) = {} & \mathsf{X}_1 \cdot \mathsf{X}_2 \cdot \ldots \cdot \mathsf{X}_k + \mathsf{X}_1 \cdot \mathsf{X}_2 \cdot \ldots \cdot \mathsf{X}_{k-1} \cdot \mathsf{X}_{k+1} + \mathsf{X}_1 \cdot \mathsf{X}_3 \cdot \ldots \cdot \mathsf{X}_{k+1} \\ & + \mathsf{X}_2 \cdot \mathsf{X}_3 \cdot \ldots \cdot \mathsf{X}_{k+1} + \mathsf{X}_2 \cdot \mathsf{X}_3 \cdot \ldots \cdot \mathsf{X}_k \cdot \mathsf{X}_{k+2} + \mathsf{X}_2 \cdot \mathsf{X}_4 \cdot \ldots \cdot \mathsf{X}_{k+2} \\ & + \ldots \\ & + \mathsf{X}_{n-l} \cdot \mathsf{X}_{n-l+1} \cdot \ldots \cdot \mathsf{X}_n \end{aligned} \tag{7.54}$$

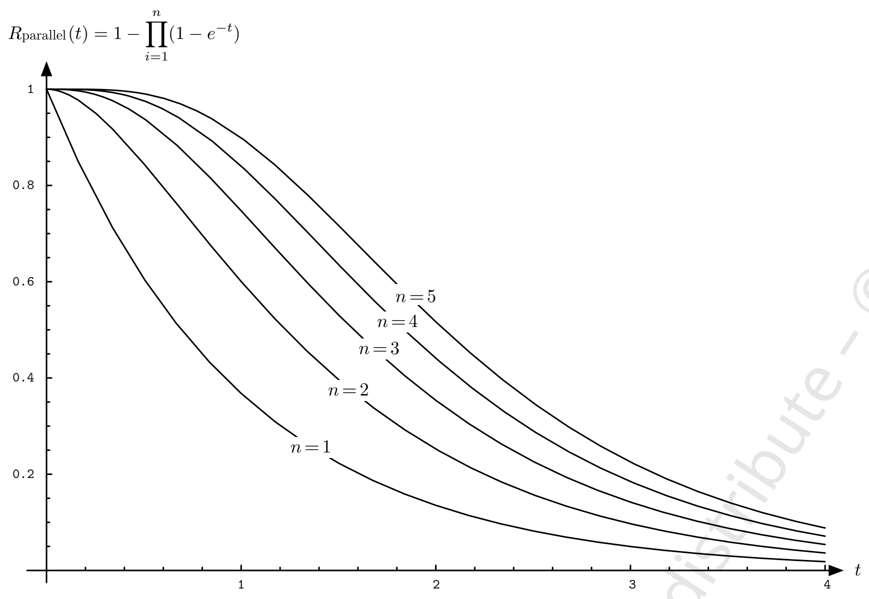


Figure 7.22: Reliability function for a parallel system without repair

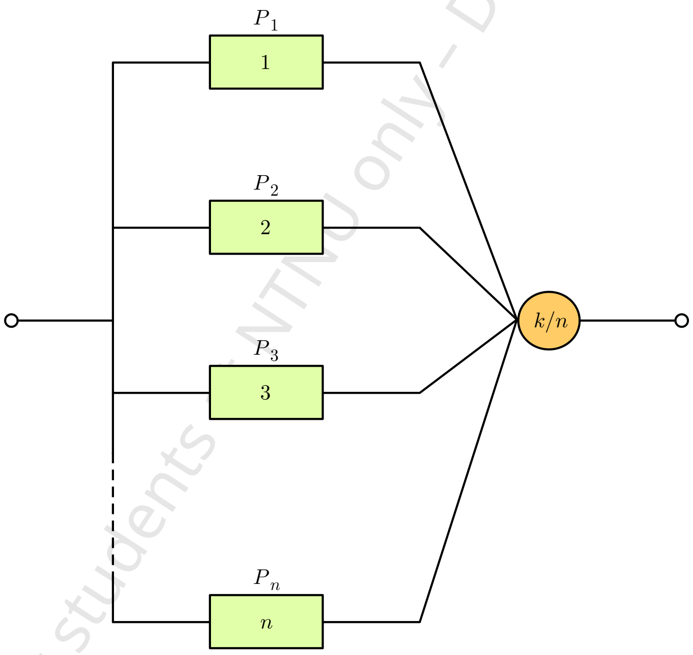


Figure 7.23: Reliability block diagram of a k-out-of-n system

Structure function of a k-out-of-n system

$$\Phi_{k-\text{out}-\text{of}-n}(\underline{\mathsf{X}}) = \sum_{\mathcal{K}_j \in \mathcal{K}} \prod_{i \in \mathcal{K}_j} \mathsf{X}_i \tag{7.55}$$

where $\mathcal{K}$ is the set of all $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ possible combinations of k out of n blocks and $\mathcal{K}_j$ is one of these sets.

For instance, in the case of a simple 2-out-of-3 system, the structure function is

$$\Phi_{2-\text{out}-\text{of}-3}(\underline{\mathsf{X}}) = \mathsf{X}_1 \cdot \mathsf{X}_2 + \mathsf{X}_1 \cdot \mathsf{X}_3 + \mathsf{X}_2 \cdot \mathsf{X}_3 \tag{7.56}$$

It is not obvious how to derive the probability $P_{k-\text{out}-\text{of}-n}$ that the system is working from the structure function. Assuming that all the subsystems are identical, $P_{\text{subsystem}} = P$, which is the case for most practical systems, the probability of a particular combination of i working subsystems and $n-i$ failed subsystems is $P^i(1-P)^{n-i}$. Hence, the probability that i of n subsystems are working is

$$\binom{n}{i} P^i (1-P)^{n-i} \tag{7.57}$$

which is recognized as the binomial distribution, see Table 2.1.

The probability that the system is working, is the sum of probabilities that k or more subsystems are working.

$$P_{k-\text{out}-\text{of}-n} = \sum_{i=k}^{n} \binom{n}{i} P^i (1-P)^{n-i} \tag{7.58}$$

The asymptotic availability is obtained by letting $P \equiv A$.

Asymptotic availability of a k-out-of-n system

$$A_{k-\text{out}-\text{of}-n} = \sum_{i=k}^{n} \binom{n}{i} A^i (1-A)^{n-i} \tag{7.59}$$

Likewise, the reliability function is obtained by letting $P \equiv R(t)$.

Reliability function of k-out-of-n system

$$R_{k-\text{out}-\text{of}-n}(t) = \sum_{i=k}^{n} \binom{n}{i} R(t)^i (1-R(t))^{n-i} \tag{7.60}$$

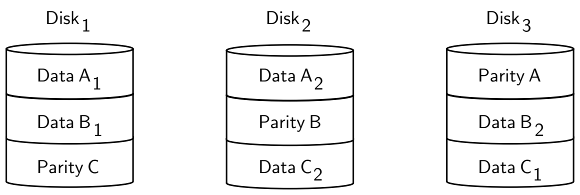


Figure 7.24: RAID level 5 with three discs

If all the subsystems in a k-out-of-n system are identical and have the same constant failure rate, $\lambda_i = \lambda$, i.e. the time to failure of every subsystem is negatively exponentially distributed with parameter λ, when there are i subsystems working, the expected time to the failure of any working subsystem is $(i\lambda)^{-1}$. It then follows that

MTFF of k-out-of-n system

$$MTFF_{k-\text{out}-\text{of}-n} = \frac{1}{\lambda}\sum_{i=k}^{n}\frac{1}{i} \tag{7.61}$$

Examples of k-out-of-n structures There are many ICT systems with k-out-of-n structures. Some typical examples are given below[5].

1. Sets of size n of subsystems with load sharing. In most cases it is sufficient that $n-1$ subsystems are working for the system to have sufficient performance. Examples from the telephony domain include tone receivers, answering machines, etc.
2. RAID (Redundant Array of Inexpensive Disks) systems. RAID systems are designed to provide highly dependable storage at a reasonable price. In a system with n disks, the data are segmented into $n-1$ data units. Each data unit is placed on a different disk and a parity segment is generated on each disk. If one disk fails, the data can be restored from the data units stored on the other disks.
3. Modular redundancy systems with majority voting. In a modular redundancy configuration, the original system is multiplied into a number of identical subsystems which are simultaneously active. Simultaneously active means that they all receive the same service requests and all deliver the same service. A mechanism, usually a majority vote, is required to decide which results are valid since the results from the subsystems are potentially different. This technique is well suited for systems with strict real-time requirements, where high reliability must be ensured during a short period relative to the *MTFF* of a subsystem. Figure 7.25 illustrates the modular redundancy principle and its performance as the number of subsystems increases.

[5]Note that in these examples both the assumption of independent repair and independent failure processes might be questionable and should be carefully considered.

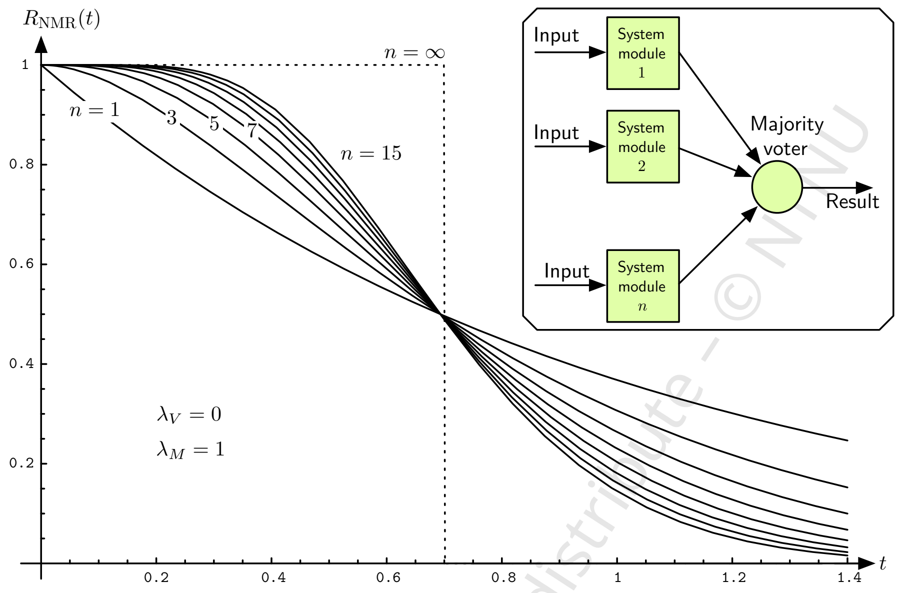


Figure 7.25: Reliability function of a n-modular redundancy system without repair

Complex structures

All the subsystems (blocks) in a reliability block diagram are assumed to be independent with respect to failures and repairs. Therefore, a complex diagram can be reduced using combinations of series, parallel, and k-out-of-n structures. All the reliability block diagrams in this book are compositions of these three basic structures.

Consider for instance the web server system shown in Figure 7.26. The system consists of the three following subsystems

1. *Network subsystem.* The web server system is connected to the network by two network servers. Each of the network servers has the capacity to handle the total workload.
2. *Bus subsystem.* A dual bus structure provides communication between the network server and storage subsystems. At least one bus must be working for the system to be working.
3. *Storage subsystem.* The storage system consists of three storage units. A storage unit includes a disk handler and a four disk RAID. Three of the four disks in the RAID system must be working, i.e. it is a 3-out-of-4 system. All three storage units must be working for the system to be working.

The objective is to determine the asymptotic availability A of this system. All the system components and subsystems are assumed to be independent with respect to failures and

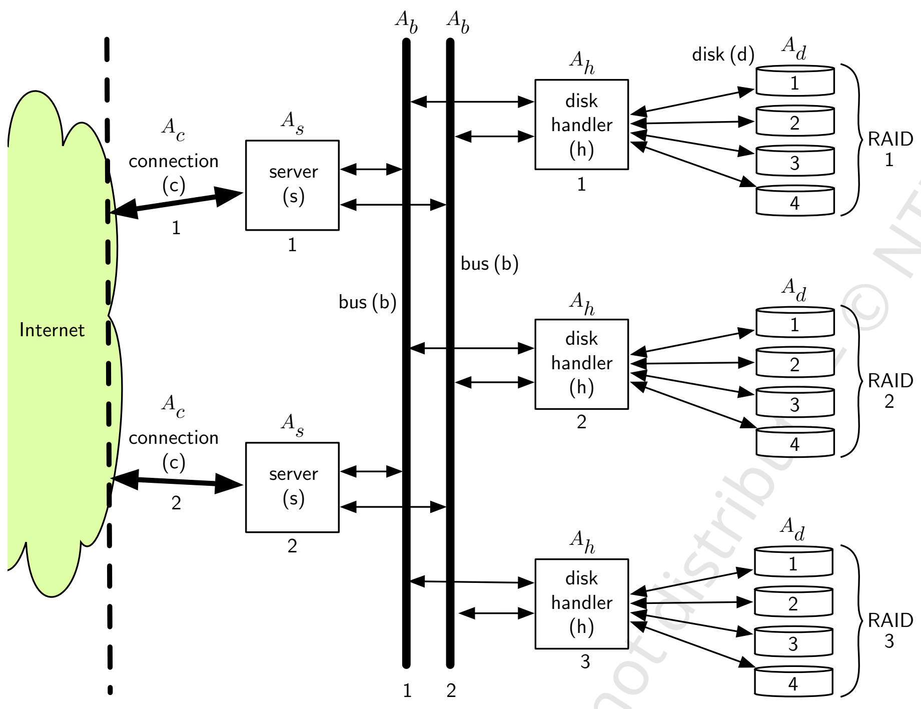


Figure 7.26: Web server system

repairs[6]. The symbols for the system components and the asymptotic availabilities are defined in Figure 7.26.

A reliability block diagram for the web server example is given in Figure 7.27. The diagram is a combination of series, parallel, and k-out-of-n structures.

The network subsystem consists of two network servers (Ns) in parallel. A network server consists in the server itself and a network connection in series. From (7.44), the availability of a network server is $A_{Ns} = A_c A_s$. The availability of the network subsystem is therefore

$$A_N = 1 - (1 - A_{Ns})^2 = 1 - (1 - A_c A_s)^2 \tag{7.62}$$

The bus subsystem also has a parallel structure. Hence, the availability of the bus subsystem is

$$A_B = 1 - (1 - A_b)^2 \tag{7.63}$$

[6]This is most likely an unrealistic assumption in a real system. Typically the number of repairmen or resources is limited, and the equipment is located in the same room or building and is therefore exposed to common external influences and faults.

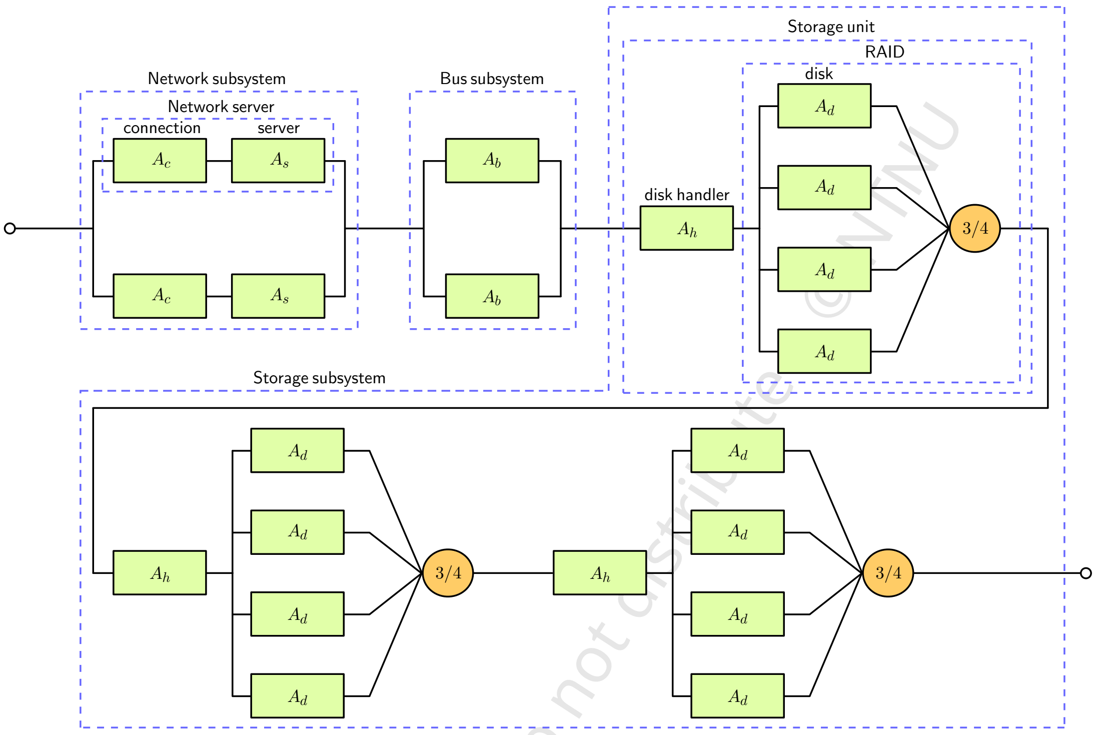


Figure 7.27: Reliability block diagram of the web server system

A storage units consists of a disk handler in series with a RAID system which is a 3-out-of-4 structure. The availability of a storage unit A_{Su} is therefore

$$A_{Su} = A_h \sum_{i=3}^{4} \binom{4}{i} A_d^i (1 - A_d)^{4-i} \tag{7.64}$$

Finally, the diagram in Figure 7.27 can be reduced to a series of one network server subsystem, one bus subsystem, and three storage unit subsystems as illustrated in Figure 7.28. The asymptotic availability of the web server system is

$$A_{\text{web}} = A_N A_B A_{Su}^3 \tag{7.65}$$

The structure function of the web server system can be obtained using a similar approach.

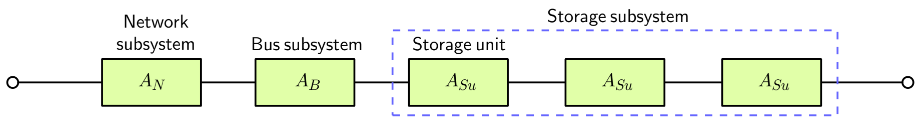


Figure 7.28: Reduced diagram of the web server system

The state of each subsystem is represented by a Boolean variable.

c_i connection $(i = 1, 2)$

s_i server $(i = 1, 2)$

b_i bus $(i = 1, 2)$

Su_i storage unit $(i = 1, 2)$

h_i disk handler $(i = 1, 2, 3)$

d_{ij} disk $(i = 1, 2, 3)$, $(j = 1, 2, 3)$

The structure functions of the network subsystem and the bus subsystem are respectively

$$\mathsf{N} = \mathsf{c}_1 \cdot \mathsf{s}_1 + \mathsf{c}_2 \cdot \mathsf{s}_2 \tag{7.66}$$

and

$$\mathsf{B} = \mathsf{b}_1 + \mathsf{b}_2 \tag{7.67}$$

The structure function of the storage unit i is

$$\mathsf{Su}_i = \mathsf{h}_i(\mathsf{d}_{i1} \cdot \mathsf{d}_{i2} \cdot \mathsf{d}_{i3} + \mathsf{d}_{i1} \cdot \mathsf{d}_{i2} \cdot \mathsf{d}_{i4} + \mathsf{d}_{i1} \cdot \mathsf{d}_{i3} \cdot \mathsf{d}_{i4} + \mathsf{d}_{i2} \cdot \mathsf{d}_{i3} \cdot \mathsf{d}_{i4}) \tag{7.68}$$

Finally, the structure function for the web server system is

$$\Phi_{\text{web}} = \mathsf{N} \cdot \mathsf{B} \cdot \prod_{i=1}^{3} \mathsf{Su}_i \tag{7.69}$$

Remember that the structure function cannot simplified based on functional identity.

7.3.2 Fault-trees

Like a reliability block diagram, a fault-tree is a graphical representation of the structure function of a system. However, while a reliability block diagram reflects the dependability structure of system components, a fault-tree focuses on *failure events* both in the system and in the environment (failure events include both faults and failures; a failure at one level becomes a fault at the next level). The strength of fault-trees is their ability to serve as a deductive tool.

Strictly speaking, a failure event is the occurrence of a failure. In the fault tree analysis terminology, it is interpreted as the event itself and the caused failure mode as long as it persists.

Failure events are combined using *logic gates* to form the *top event.* A logic gate performs a simple logical function. The top event is the main failure to be modeled and analyzed; for ICT-systems this is typically a system failure, i.e. the system does not deliver the service in accordance to the specification. At the opposite, a leaf node is the "lowest level" event in the model. We distinguish two types of leaf nodes:

- *Primary events.* Basic events in the system, see Section 7.1.1. Note that it is not always obvious whether a failure is a basic event or if it can be broken down into more details. What is defined as a primary event depends on the modeling and analysis objectives and on at which level models and/or data for failure rates and repair are available.
- *Non-expanded events.* Events that are not (yet) further expanded or broken down into more details. This is either because details are missing, or because the failure is considered to have an insignificant contribution to the top event.

A fault-tree may include the same leaf node in several sub-trees. This implies that the probability of the top event cannot be obtained by performing successive reductions of the model as in the case of reliability block diagrams, and more advanced techniques are needed.

Negated structure function

The state of a system component is now regarded as a Boolean variable Y_i defined as

$$\begin{aligned} X_i = W \Leftrightarrow \mathsf{Y}_i = \text{False} \\ X_i = F \Leftrightarrow \mathsf{Y}_i = \text{True} \end{aligned} \tag{7.70}$$

The corresponding boolean system state vector is $\underline{\mathsf{Y}} = \{\mathsf{Y}_1, \ldots, \mathsf{Y}_n\}$.

The negated structure function $\Psi(\underline{\mathsf{Y}})$ of a system is a Boolean function defined as

Negated structure function

$$\Psi(\underline{\mathsf{Y}}) = \begin{cases} \text{True} & \text{if } \underline{X} \in \Omega_F \\ \text{False} & \text{if } \underline{X} \in \Omega_W \end{cases} \tag{7.71}$$

In other words, $\Psi(\underline{\mathsf{Y}})$ is true if the system is failed, false otherwise.

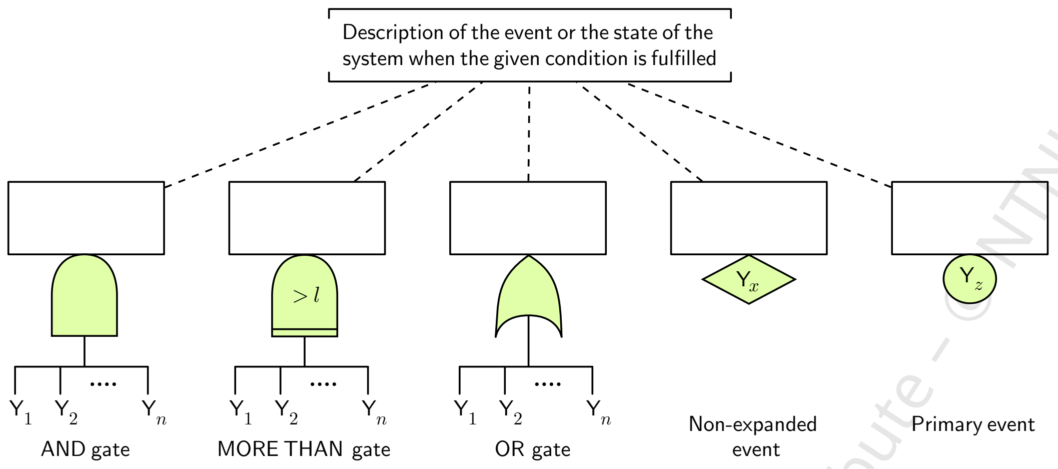


Figure 7.29: Symbols in fault-trees

Symbols

The symbols for the AND, OR, and MORE THAN logic gates are depicted in Figure 7.29. Some textbooks and fault-tree tools extend the modeling power of fault-trees by introducing additional symbols and notations, extra functionality, and "meta" symbols to break large trees into subtrees and to allow reuse of sub-structures in different parts of the fault-tree.

A fault-tree is constructed top-down, but the effects of the events propagate bottom-up. The input to a logic gate is below the symbol, the output, i.e. the result of the logical function performed by the gate, on the top of the gate symbol. The symbols are similar to those used to represent digital circuits.

Example 7.4 Model of a PC
A non-fully expanded fault-tree for the PC system described in Example 7.2 is given in Figure 7.30.

AND **logic gate**

The output event of an AND logic gate is present when all the n input events Y_i are present simultaneously, i.e. the gate performs a logical AND.

AND **function**

$$\Psi_{\mathsf{AND}}(\underline{\mathsf{Y}}) = \mathsf{Y}_1 \cdot \mathsf{Y}_2 \cdot \dots \cdot \mathsf{Y}_n = \prod_{i=1}^{n} \mathsf{Y}_i \qquad (7.72)$$

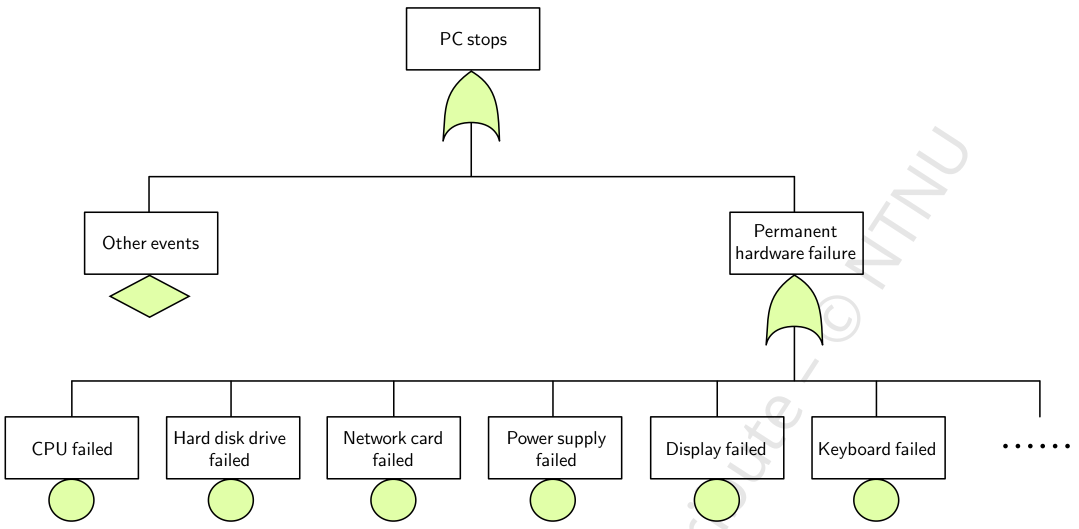


Figure 7.30: Not fully expanded fault tree for a PC

OR **logic gate**

The output event of a OR logic gate is present if any of the n input events Y_i is present, i.e. the gate performs a logical OR.

OR function

$$\Psi_{OR}(\underline{Y}) = Y_1 + Y_2 + \cdots + Y = \sum_{i=1}^{n} Y_i \tag{7.73}$$

MORE THAN **logic gate**

The output event of a MORE THAN logic gate is present when more than the given number l of the n input events are present simultaneously, $0 \leqslant l < n$. When $l = 0$ the MORE THAN logic gate is a OR gate, when $l = n - 1$ it is a AND logic gate.

The logical function performed by a MORE THAN includes all the possible combinations of $l + 1$ failure events.

$$\begin{aligned}\Psi_{MORE\,THAN}(\underline{Y}) = {} & Y_1 \cdot Y_2 \cdot \ldots \cdot Y_{l+1} + Y_1 \cdot Y_2 \cdot \ldots \cdot Y_l \cdot Y_{l+2} + Y_1 \cdot Y_3 \cdot \ldots \cdot Y_{l+2} \\ & + Y_2 \cdot Y_3 \cdot \ldots \cdot Y_{l+2} + Y_2 \cdot Y_3 \cdot \ldots \cdot Y_{l+1} \cdot Y_{l+3} + Y_2 \cdot Y_4 \cdot \ldots \cdot Y_{l+3} \\ & + \ldots \\ & + Y_{n-l} \cdot Y_{n-l+1} \cdot \ldots \cdot Y_n\end{aligned} \tag{7.74}$$

MORE THAN function

$$\Psi_{\text{MORE THAN}}(\underline{\mathsf{Y}}) = \sum_{\mathcal{L}_j \in \mathcal{L}} \prod_{i \in \mathcal{L}_j} \mathsf{Y}_i \tag{7.75}$$

where $\mathcal{L}$ is the set of all $\binom{n}{l+1}$ possible combinations of $l+1$ out of n events and $\mathcal{L}_j$ is one of these sets.

For example, with $n = 3$ and $l = 1$ (2-out-of-3 structure) the set of combinations is $\mathcal{L} = \{\{1,2\},\{1,3\},\{2,3\}\}$ and the logical function performed by the MORE THAN logic gate is $\mathsf{Y}_1 \cdot \mathsf{Y}_2 + \mathsf{Y}_1 \cdot \mathsf{Y}_3 + \mathsf{Y}_2 \cdot \mathsf{Y}_3$

Note that the MORE THAN logic gate is not a standard, or basic, gate in fault-trees. However using it significantly simplifies the fault-trees compared to fault-trees consisting of AND and OR logic gates only. As an illustration, a 2-out-of-3 structure is represented both using the MORE THAN logic gate and using only AND and OR logic gates in Figure 7.31. The larger the k-out-of-n structure, the larger the reduction in terms of number of leaf nodes. For instance, in a 4-out-of-12 structure the fault-tree with AND and OR logic gates only contains $\binom{12}{4} = 792$ leaf nodes, while a fault-tree with MORE THAN gates has only 12.

Complex structures

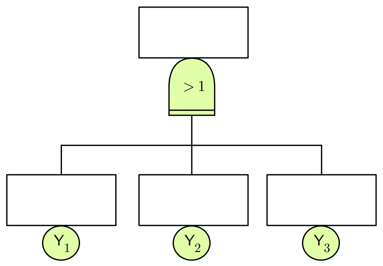
Complex structures are modeled top-down by deduction through successive expansions of the failure events in the system.

For instance, a partially expanded fault-tree of the web server example illustrated in Figure 7.26 is given in Figure 7.32. It shows how to map the dependability structure of a system to a fault-tree. It also shows that it is not easy to refind the structure function from a fault-tree.

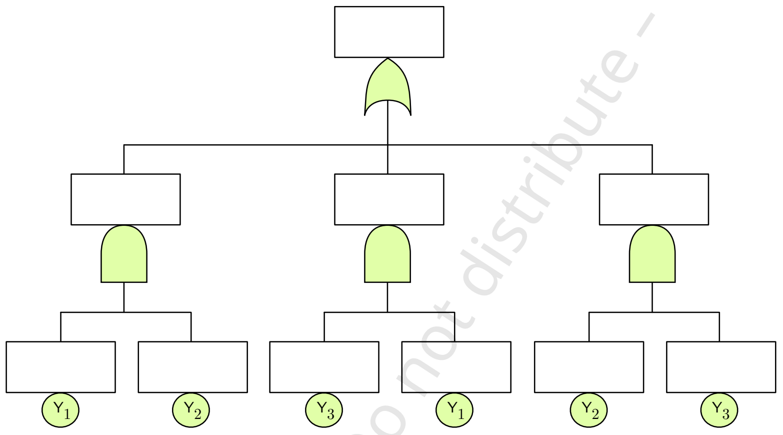
7.3.3 Correspondence between reliability block diagrams and fault-trees

Modeling based on reliability block diagrams is a constructive approach with focus on working system components, X_i. It reflects the dependability structure of the system and the structure function $\Phi(\underline{\mathsf{X}})$ of the system can easily be derived from a reliability block diagram. The structure function expresses whether the system is working or not for all combinations of states of the system components.

Modeling based on fault-trees is a deductive approach with focus on failure events, $\mathsf{Y}_i = \bar{\mathsf{X}}_i$. The logical function represented by a fault-tree is the negated structure function $\Psi(\underline{\mathsf{Y}})$ of the system. The negated structure function expresses whether the system is failed or not for all combinations of failures events.



(a) With a MORE THAN gate



(b) With AND and OR gates only

Figure 7.31: Alternative representations of a 2-out-of-3 structure using a fault-tree

Using De Morgan's theorem gives

$$\Phi(\underline{X}) = \overline{\Psi(\underline{Y})},\ \underline{X} = \underline{\bar{Y}} \tag{7.76}$$

There is an unique mapping between the structure function and the negated structure function and therefore also between the reliability block diagram representation and the fault-tree representation of the dependability structure of a system.

Figure 7.33 gives the correspondence between the representations using reliability block diagrams and fault-trees and the structure functions and negated structure functions for series, parallel and k-out-of-n structures.

Note that the negated structure function for a k-out-of-n structure given in Figure 7.33 is obtained from the corresponding structure function (7.55) using (7.76). It is logically equivalent to (7.75) but their forms differ; one is given as a product of sums, the other as a sum of products. Generally, a structure function may be written in either form and, using Boolean algebra, it is possible to transform one form to the other. This is discussed in the next section.

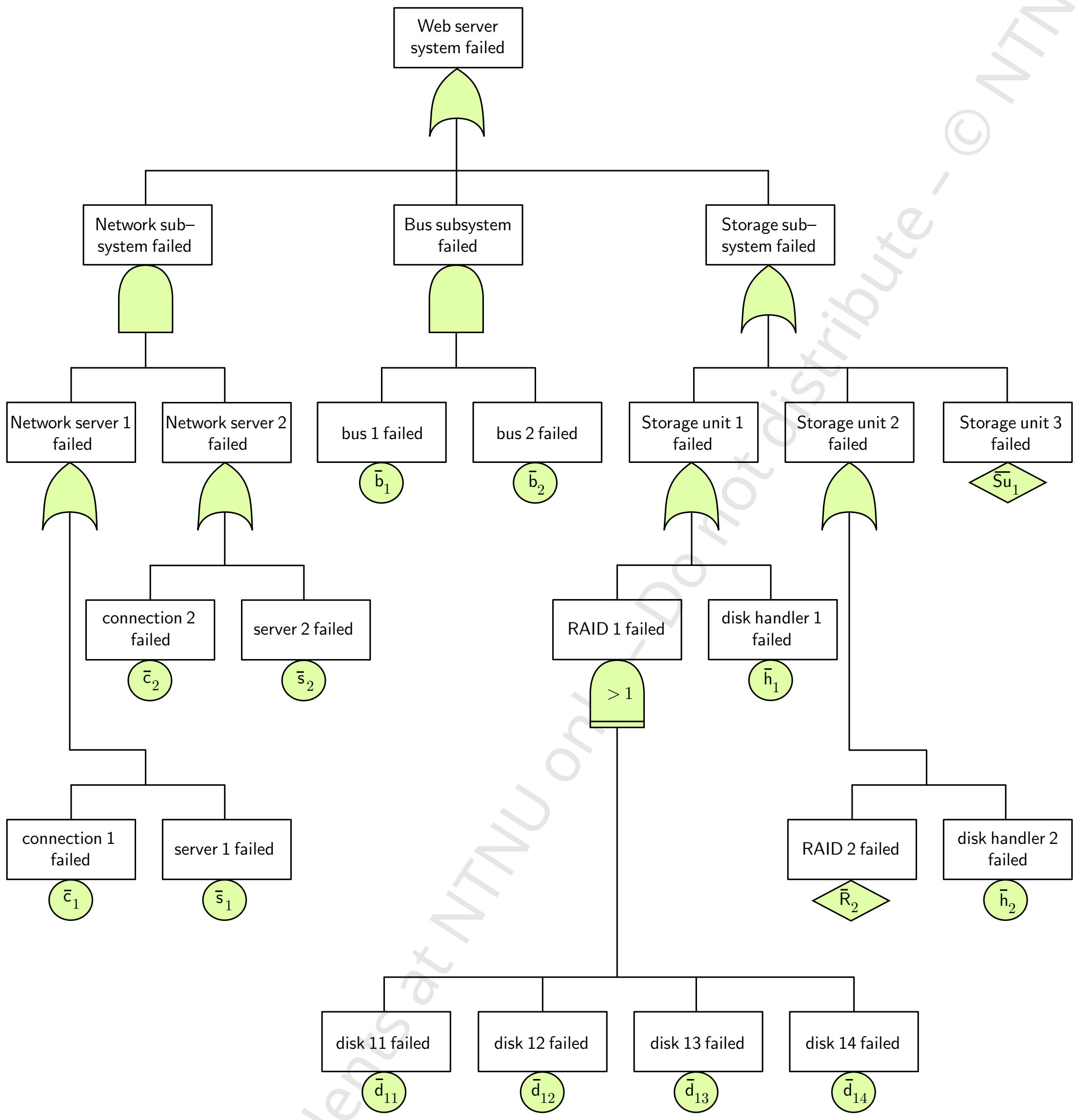


Figure 7.32: Fault-tree model of the web server example in Figure 7.26.

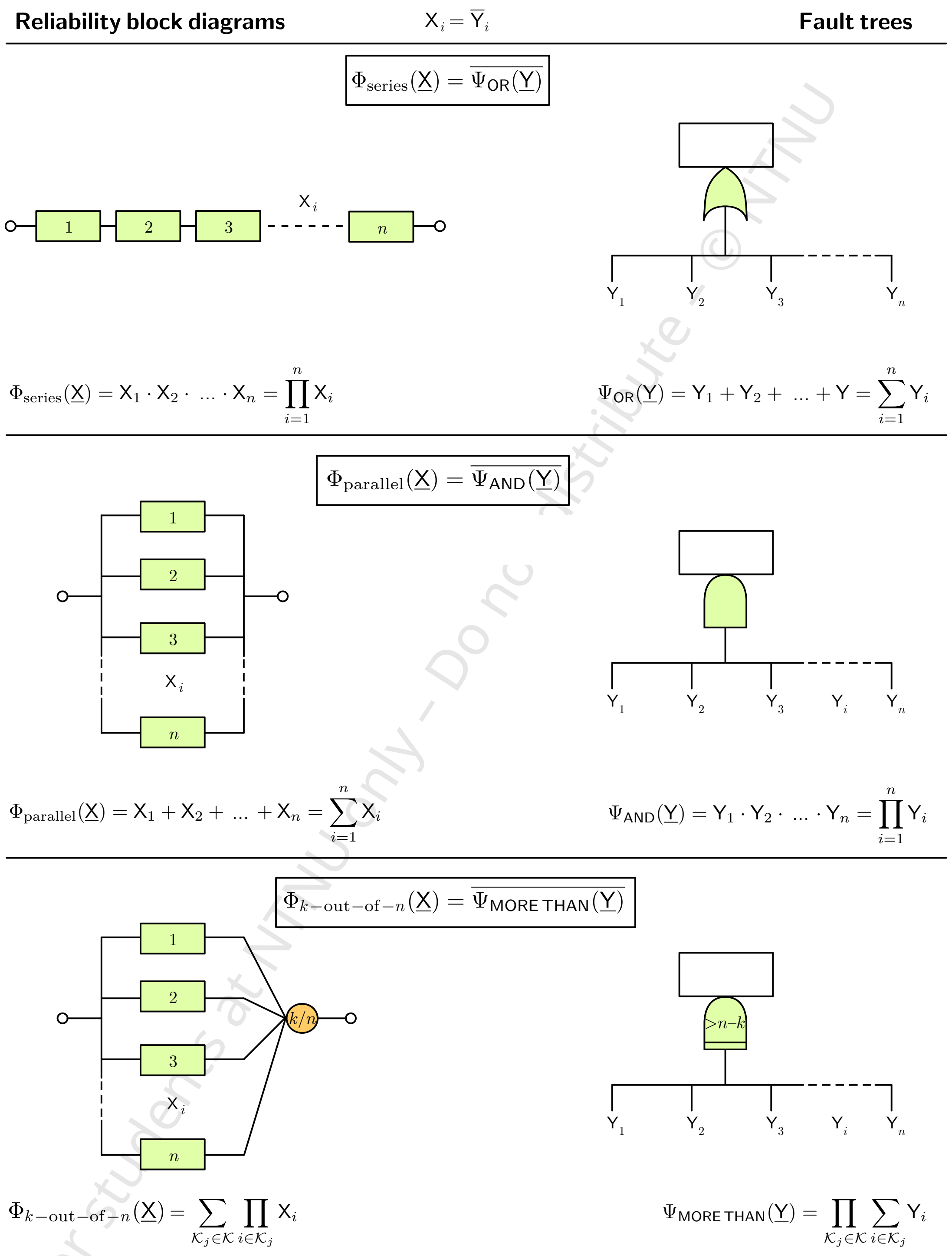

Reliability block diagrams
$X_i = \overline{Y}_i$
Fault trees
$\Phi_{\text{series}}(\underline{X}) = \overline{\Psi_{\text{OR}}(\underline{Y})}$
1
2
3
X_i
n
Y_1
Y_2
Y_3
Y_n
$\Phi_{\text{series}}(\underline{X}) = X_1 \cdot X_2 \cdot \; \dots \cdot X_n = \prod_{i=1}^{n} X_i$
$\Psi_{\text{OR}}(\underline{Y}) = Y_1 + Y_2 + \; \dots + Y = \sum_{i=1}^{n} Y_i$
$\Phi_{\text{parallel}}(\underline{X}) = \overline{\Psi_{\text{AND}}(\underline{Y})}$
1
2
3
X_i
n
Y_1
Y_2
Y_3
Y_i
Y_n
$\Phi_{\text{parallel}}(\underline{X}) = X_1 + X_2 + \; \dots + X_n = \sum_{i=1}^{n} X_i$
$\Psi_{\text{AND}}(\underline{Y}) = Y_1 \cdot Y_2 \cdot \; \dots \cdot Y_n = \prod_{i=1}^{n} Y_i$
$\Phi_{k-\text{out}-\text{of}-n}(\underline{X}) = \overline{\Psi_{\text{MORE THAN}}(\underline{Y})}$
1
2
3
X_i
n
k/n
$>n-k$
Y_1
Y_2
Y_3
Y_i
Y_n
$\Phi_{k-\text{out}-\text{of}-n}(\underline{X}) = \sum_{\mathcal{K}_j \in \mathcal{K}} \prod_{i \in \mathcal{K}_j} X_i$
$\Psi_{\text{MORE THAN}}(\underline{Y}) = \prod_{\mathcal{K}_j \in \mathcal{K}} \sum_{i \in \mathcal{K}_j} Y_i$

Figure 7.33: Correspondence between reliability block diagrams and fault-trees

7.3.4 Qualitative dependability analysis

This section briefly presents techniques for qualitative dependability analysis based on structural models.

As mentioned in the previous section, the structure function of a system is a logical function that expresses whether the system is working or not. As a boolean function, any structure function can be expressed in a canonical form.

Minimal sum-of-products form (first canonical form)

A minimal sum-of-products is an irreducible Boolean sum (logical OR) of minterms, where a minterm is a Boolean product (logical AND) that may include a variable only once, e.g. $\mathsf{ABC} + \mathsf{ABD} + \mathsf{BE}$.

Minimal product-of-sums form (second canonical form)

A minimal product-of-sums is an irreducible Boolean product (logical AND) of max-terms, where a maxterm is a Boolean sum (logical OR) that may include a variable only once, e.g. $(\mathsf{A} + \mathsf{E}) \cdot (\mathsf{C} + \mathsf{D} + \mathsf{E}) \cdot \mathsf{B}$.

Key dependability properties can be obtained from structure functions expressed in a canonical form, namely path and cut sets. Path sets focus on identifying combinations of working subsystems that are required for the system to work (designer's point of view). Cut sets focus on combinations of failed subsystems/elements that make the system fail (saboteur's point of view). Reliability block diagrams constituted exclusively of a series and parallel structures can be mapped to (reliability) graphs [44] where the reliability blocks correspond to branches in the graph. The minimal path and cut sets from reliability block structures correspond then to the same concepts as in graph theory.

Cut set

The system is failed if all the subsystems in a cut set are failed.

Minimal cut set

The system is failed if and only if all the subsystems in a minimal cut set are failed, given that all the other subsystems not in the set are working.

If all the subsystems in a system have an unavailability in the same order of magnitude (e.g. 10^{-3}), the subsystems in the cut sets with the fewest number of elements (the lowest cardinality) dominate the system unavailability.

Path set

The system is working if all the subsystems in a path set are working.

Minimal path set

The system is working if and only if all the subsystems in a minimal path set are working, and given that all the subsystems not in the set are failed.

A structure function expressed in minimal product-of-sums form is of form:

$$\Phi(\underline{X}) = (X_i + \ldots + X_j) \cdot (X_k + \ldots + X_l) \cdot \ldots \cdot (X_m + \ldots + X_n) \tag{7.77}$$

As long as at least one of the maxterms of the form $(X_k + \ldots + X_l)$ in $\Phi(\underline{X})$ is False then the structure function is False and the system is failed. Hence,

Structure function expressed in minimal product-of-sums form

Each maxterm of the structure function expressed in a minimal product-of-sums form corresponds to a *minimal cut set*.

A structure function expressed in minimal sum-of-products form is of form

$$\Phi(\underline{X}) = (X_i \cdot \cdots \cdot X_j) + (X_k \cdot \cdots \cdot X_l) + \cdots + (X_m \cdot \cdots \cdot X_n) \tag{7.78}$$

As long as at least one of the minterms of the form $(X_k \cdot \cdots \cdot X_l)$ in $\Phi(\underline{X})$ is True then the structure function is True and the system is working. Hence,

Structure function in minimal sum-of-products form

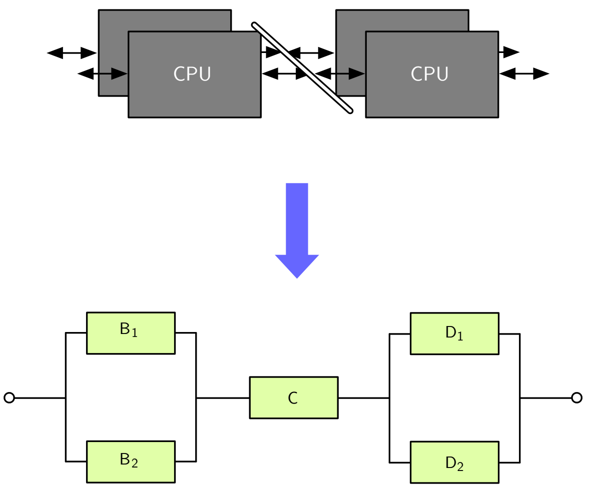
Each minterm of the structure function expressed in minimal sum-of-products form corresponds to a *minimal path set*.

Example 7.5 Path and cut sets
Examples of path and cut sets in the case of a simple series-parallel system are given in Figure 7.34. Minimal cut and path set can be found from the structure function written in minimal product-of-sums and sum-of-products forms.

$$\Phi(\underline{X}) = (B_1 + B_2)C(D_1 + D_2) = B_1CD_1 + B_1CD_2 + B_2CD_1 + B_2CD_2$$

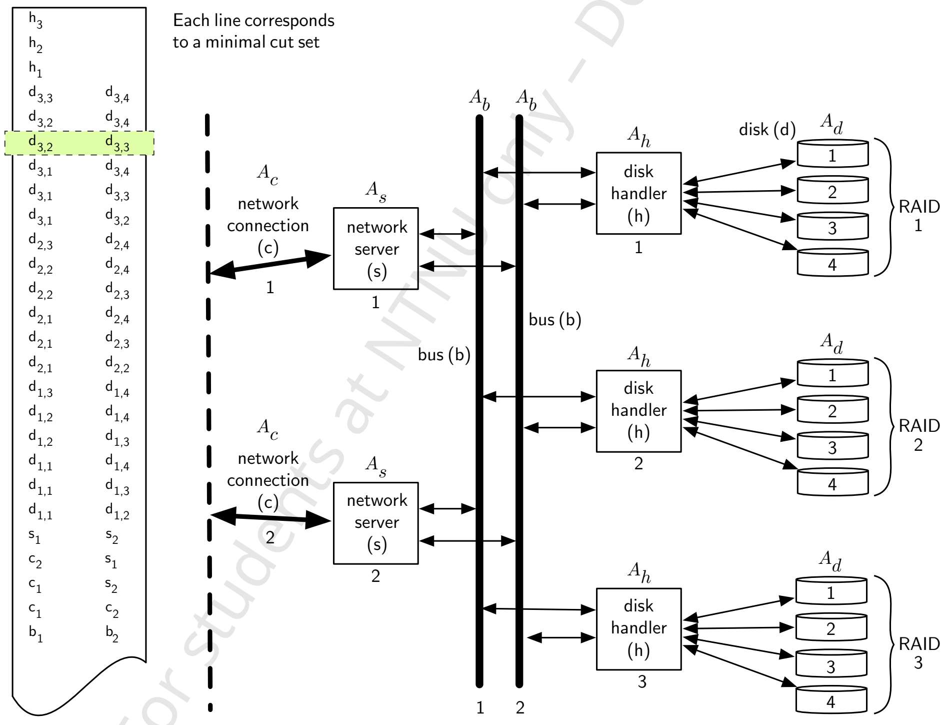
The minimal cut sets are the maxterms of the minimal product-of-sums form. The minimal path sets are the minterms of the minimal sum-of-products form.
This example illustrates the mapping between minimal path and cut sets and the subsystems in the model. The minimal cut set $\{B_1, B_2\}$ means that if both B_1 and B_2 are failed the system is failed. Likewise, the minimal path set $\{B_1, C, D_1\}$ means that if B_1, C and D_1 are working, the system is working, regardless the state of the other system components.

Example 7.6 Minimal cut and path sets for the web server system
The minimal cut sets are given in Figure 7.35 and the minimal path sets in Figure 7.36 for the web server system presented in Figure 7.26.



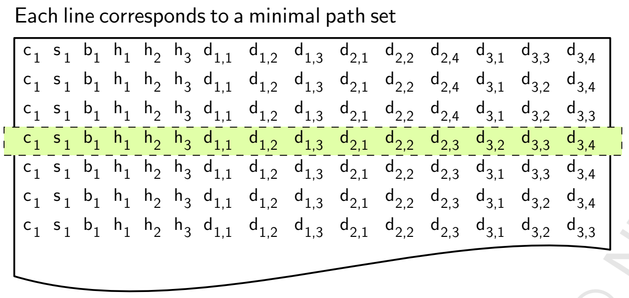
Cut sets	Path sets
B_1 B_2 C D_1 D_2	B_1 B_2 C D_1 D_2
B_1 C D_1 D_2	B_1 C D_1 D_2
B_2 C D_1 D_2	B_2 C D_1 D_2
B_1 B_2 C D_1	B_1 B_2 C D_1
B_1 B_2 C D_2	B_1 B_2 C D_2
B_1 B_2 D_1 D_2	**B_2 C D_1**
B_1 D_1 D_2	**B_1 C D_1**
B_2 D_1 D_2	**B_2 C D_2**
B_1 B_2 D_1	**B_1 C D_2**
B_1 B_2 D_2	
C D_1 D_2	
B_2 C D_1	
B_1 C D_1	
B_2 C D_2	
B_1 C D_2	
C D_2	
C D_1	
B_2 C	
B_1 C	
B_1 B_2	
D_1 D_2	
C	

Figure 7.34: Reliability block diagrams and corresponding cut and path sets (minimal cut and path sets are in bold font)



h_3	
h_2	
h_1	
$d_{3,3}$	$d_{3,4}$
$d_{3,2}$	$d_{3,4}$
$d_{3,2}$	$d_{3,3}$
$d_{3,1}$	$d_{3,4}$
$d_{3,1}$	$d_{3,3}$
$d_{3,1}$	$d_{3,2}$
$d_{2,3}$	$d_{2,4}$
$d_{2,2}$	$d_{2,4}$
$d_{2,2}$	$d_{2,3}$
$d_{2,1}$	$d_{2,4}$
$d_{2,1}$	$d_{2,3}$
$d_{2,1}$	$d_{2,2}$
$d_{1,3}$	$d_{1,4}$
$d_{1,2}$	$d_{1,4}$
$d_{1,2}$	$d_{1,3}$
$d_{1,1}$	$d_{1,4}$
$d_{1,1}$	$d_{1,3}$
$d_{1,1}$	$d_{1,2}$
s_1	s_2
c_2	s_1
c_1	s_2
c_1	c_2
b_1	b_2

Figure 7.35: Minimal cut sets of the web server system example


Each line corresponds to a minimal path set

$c_1\ s_1\ b_1\ h_1\ h_2\ h_3\ d_{1,1}\ d_{1,2}\ d_{1,3}\ d_{2,1}\ d_{2,2}\ d_{2,4}\ d_{3,1}\ d_{3,3}\ d_{3,4}$

$c_1\ s_1\ b_1\ h_1\ h_2\ h_3\ d_{1,1}\ d_{1,2}\ d_{1,3}\ d_{2,1}\ d_{2,2}\ d_{2,4}\ d_{3,1}\ d_{3,2}\ d_{3,4}$

$c_1\ s_1\ b_1\ h_1\ h_2\ h_3\ d_{1,1}\ d_{1,2}\ d_{1,3}\ d_{2,1}\ d_{2,2}\ d_{2,4}\ d_{3,1}\ d_{3,2}\ d_{3,3}$

$c_1\ s_1\ b_1\ h_1\ h_2\ h_3\ d_{1,1}\ d_{1,2}\ d_{1,3}\ d_{2,1}\ d_{2,2}\ d_{2,3}\ d_{3,2}\ d_{3,3}\ d_{3,4}$

$c_1\ s_1\ b_1\ h_1\ h_2\ h_3\ d_{1,1}\ d_{1,2}\ d_{1,3}\ d_{2,1}\ d_{2,2}\ d_{2,3}\ d_{3,1}\ d_{3,3}\ d_{3,4}$

$c_1\ s_1\ b_1\ h_1\ h_2\ h_3\ d_{1,1}\ d_{1,2}\ d_{1,3}\ d_{2,1}\ d_{2,2}\ d_{2,3}\ d_{3,1}\ d_{3,2}\ d_{3,4}$

$c_1\ s_1\ b_1\ h_1\ h_2\ h_3\ d_{1,1}\ d_{1,2}\ d_{1,3}\ d_{2,1}\ d_{2,2}\ d_{2,3}\ d_{3,1}\ d_{3,2}\ d_{3,3}$

Figure 7.36: 7 of the 256 path sets for the web server system example

So far, only structure functions and reliability block diagrams have been considered. Cut and path sets can also be derived from negated structure functions and fault-trees. For instance, focusing on failure events rather than on system components, a cut set is defined as the set of events whose simultaneous presence results in the presence of the top event (system failure) [45]. Also, from (7.76) and using De Morgan's theorem, the maxterms of a negated structure function expressed in product-of-sums form correspond to minimal path sets and the minterms of a negated structure function expressed in minimal sum-of-products form correspond to minimal cut sets.

Note that, the results presented in this section are not only valid for series-parallel structures, but are generally applicable for systems with more complex structures. The minimal cut and path sets, and the corresponding minimal product and sum forms of the structure function are recommended starting points for (approximated) quantitative dependability analysis of a system. See [46, 45] for more details.

7.4 Combination of Markov and structural models

Two dependability modeling approaches have been presented in this chapter, namely Markov models (Section 7.2) and structural models (Section 7.3). On one hand, Markov models allow to capture the dependencies in the subsystem behavior, multiple failure modes (e.g. permanent hardware, transient hardware, and software failures) and sequence of events but lead to large and complex state transition diagrams (state explosion). On the other hand, structural models allow to model larger systems based on their structure but ignore to a great extent the system dynamics and dependencies. To model large systems with dynamics, several failure modes, and some dependencies between subsystems, a combination of structural models and Markov models is a viable approach as described in the following.

7.4.1 Independent subsystems view

A system can be modeled by a reliability block diagram where each block corresponds to a subsystem. The subsystems are assumed statistically independent of each other. The detailed behavior of each subsystem can be modeled by a state transition diagram and the dependability properties of the system can be obtained by combining the corresponding properties of all the subsystems

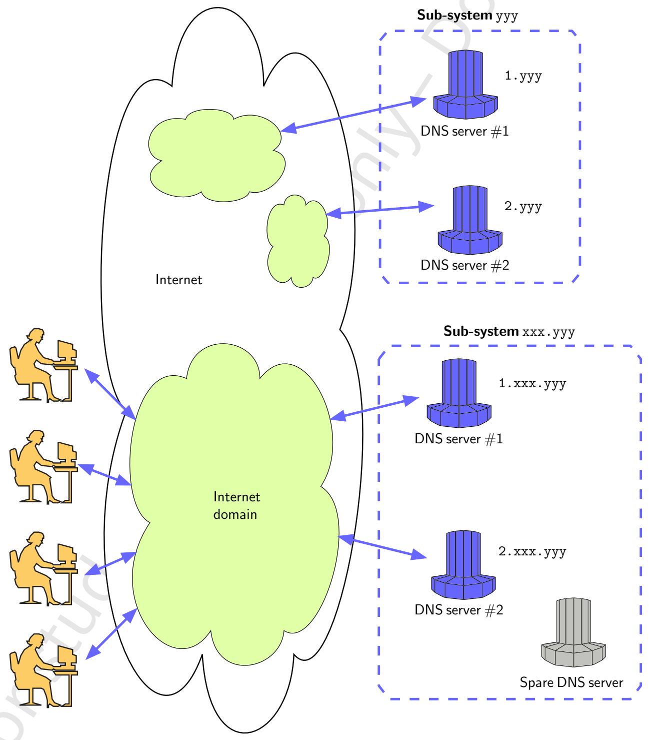
For each subsystem i, the Boolean variable X_i is defined as

$$\begin{aligned} X_i \in \Omega_{W_i} &\Leftrightarrow \mathsf{X}_i = \mathsf{True} \\ X_i \in \Omega_{F_i} &\Leftrightarrow \mathsf{X}_i = \mathsf{False} \end{aligned} \tag{7.79}$$

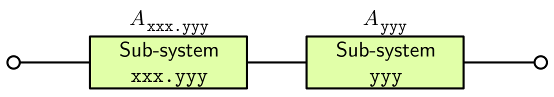
where $\Omega_{W_i} \in \Omega_i$ is the set of working states of the subsystem i and $\Omega_{F_i} \in \Omega_i$ the set of failed states.

The structure function of the entire system is $\Phi(\mathsf{X}_1, \dots, \mathsf{X}_n)$ where n is the number of subsystems.

Example 7.7 Extended DNS server model
The model of the DNS server system in Section 7.2.3 is extended to include a two level DNS functionality as illustrated in the figure below.



The DNS service is operational and working when address resolution (lookup) is working for addresses of its own domain, `xxx.yyy`, and higher level addresses `yyy`. This means that the DNS server system must be working at both levels to be working. The corresponding reliability block diagram is:



The two DNS server systems (subsystem `xxx.yyy` and `yyy`) are not geographically collocated and are operated and maintained by different organizations. The assumption about statistical independence between the subsystems is therefore reasonable. Furthermore, for the sake of simplicity, only permanent failures are included in the model. The servers in the two subsystems have the same failure and repair rates. Each subsystem is modeled by a reliability block and the system is modeled by a series structure of the two reliability blocks.
The asymptotic availability of each block is obtained from Markov models. DNS domain `xxx.yyy` has no standby server and the asymptotic availability of this domain, $A_{\texttt{xxx.yyy}}$, is obtained from the Markov model in Figure 7.10. The asymptotic availability $A_{\texttt{yyy}}$ of subsystem `yyy` is obtained from the Markov model in Figure 7.13.
Finally, the availability of the entire DNS system is obtained as

$$A_{\text{DNS}} = A_{\texttt{xxx.yyy}} A_{\texttt{yyy}} = 1 - (1 - U_{\texttt{xxx.yyy}})(1 - U_{\texttt{yyy}})$$

where $U_{\texttt{yyy}}$ is determined using (7.12) and $U_{\texttt{xxx.yyy}}$ using (7.20).
Obtaining the availability from a Markov model of the two subsystems `xxx.yyy` and `yyy` would have been much harder. A Markov model of the entire system has $3 \cdot 7 = 21$ states instead of only $3 + 7 = 10$ states when using two separate models.

7.4.2 Collective subsystems view

In the previous section, the state of the entire system only depends on the state of the subsystem, i.e. the structure function, see Section 7.3.1, of the entire system is a function of the boolean state variables of the subsystems only. This approach can be generalized by letting the structure function depend directly on the states of the subsystems. In other words, the subsystems are still assumed to be statistically independent, but the definition of working or failed state for the entire system may now depend on a combination of (internal) states in the different subsystems.

For a subsystem i, the Boolean state variable X_{ij} is defined as

$$\begin{aligned} X_i(t) = j \Leftrightarrow \mathsf{X}_{ij} = \mathsf{True} \\ X_i(t) \neq j \Leftrightarrow \mathsf{X}_{ij} = \mathsf{False} \end{aligned} \tag{7.80}$$

The structure function Φ of the entire system is defined as

$$\begin{aligned} \{j_1, \ldots, j_n\} \in \Omega_W \Leftrightarrow \Phi\left(\underline{\mathsf{X}}_1^{(j_1)}, \ldots, \underline{\mathsf{X}}_n^{(j_n)}\right) = \mathsf{True} \\ \{j_1, \ldots, j_n\} \in \Omega_F \Leftrightarrow \Phi\left(\underline{\mathsf{X}}_1^{(j_1)}, \ldots, \underline{\mathsf{X}}_1^{(j_n)}\right) = \mathsf{False} \end{aligned} \tag{7.81}$$

where n is the number of subsystems and

$$\begin{aligned} \underline{\mathsf{X}}_i^{(j_i)} &= \{\mathsf{X}_{i1}, \cdots, \mathsf{X}_{im_i}\} \\ \mathsf{X}_{ij} &= \begin{cases} \mathsf{True} & \text{if } j = j_i \\ \mathsf{False} & \text{if } j \neq j_i \end{cases} \end{aligned} \tag{7.82}$$

and m_i is the number of states of subsystem i.

Since a subsystem can only be in one state at a time and the subsystems are assumed independent,

$$P_{\text{system}} = \sum_{j_1=1}^{m_1} p_{1j_1} \sum_{j_2=1}^{m_2} p_{2j_2} \cdots \sum_{j_n=1}^{m_n} p_{nj_n} I\left(\Phi\left(\underline{\mathsf{X}}_1^{(j_1)}, \ldots, \underline{\mathsf{X}}_1^{(j_n)}\right)\right) \tag{7.83}$$

where $p_{ij} = \lim_{t\to\infty} P(X_i(t) = j)$ is the asymptotic probability that the subsystem i is in state j and I denotes an indicator function ($I(x) = 1$ if $x =$ True and $I(x) = 0$ if $x =$ False).

Example 7.8 Extended DNS server model (continued)
Assume now that the DNS system is working (i.e. has sufficient capacity) if and only if at least three servers in total are working. Let us index the subsystem yyy with 1 and the subsystem `xxx.yyy` with 2. The the structure function for system is then given by the following Boolean function

$$\Phi(\underline{\mathsf{X}}_1, \underline{\mathsf{X}}_2) = \mathsf{X}_{11} \cdot (\mathsf{X}_{21} + \mathsf{X}_{22} + \mathsf{X}_{27} + \mathsf{X}_{28} + \mathsf{X}_{29}) + \mathsf{X}_{12} \cdot (\mathsf{X}_{21} + \mathsf{X}_{27} + \mathsf{X}_{29})$$

where the state indexes for each subsystem correspond to the state numbers used in Figures 7.10 and 7.13, respectively.
Using (7.83), the availability of the system becomes

$$A_{\mathsf{DNS}} = p_{11} \cdot (p_{21} + p_{22} + p_{27} + p_{28} + p_{29}) + p_{12} \cdot (p_{21} + p_{27} + p_{29})$$

8 Experimental study of stochastic processes

During a simulation study, see Chapter 4, or a measurement study on a real system, values of parameters indicating the performance or the dependability are observed and recorded, e.g. the filling level of a buffer, the time to first failure, the call duration, the number of failures during the first year of operation, the number of entries in a database, the traffic on a link, the utilization of a processor etc. Each of these values are observations of a stochastic process. The characteristics of the studied process are often unknown, but the process is a function of one or more random variables and the observations themselves are therefore random variables.

In this chapter, Section 8.1 describes how to observe a stochastic process, i.e. how to collect a set of observations. Section 8.2 details how to analyze experimental data, i.e. how to derive estimates of parameters of interest from a set of observations, and how to quantify the uncertainty on these estimates. Finally, Section 8.3 addresses briefly what information to record during an experiment and how to organize an experiment to obtain independent and identically normally distributed observations in the (common) case where direct measurement results neither in independent nor normally distributed observations.

8.1 Observation of a stochastic process

Let $X(t)$ be a stochastic process like the one shown in Figure 8.1. This process can for instance represent the number of bytes in a buffer or the availability of a system. There are two ways of observing such a process $X(t)$; either continuously (*continuous observation*) or discretely (*sampling*). These methods apply both for simulated and real processes, and are illustrated by the shaded area and the vertical dashed lines in Figure 8.1, respectively.

8.1.1 Continuous observation

Continuously observing a process consists in recording the state of a process at all times. As a result, for example, the *time average* can be estimated.

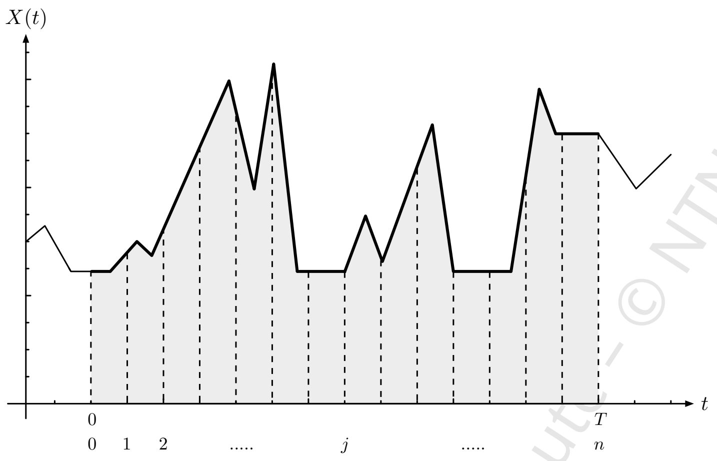


Figure 8.1: Continuous time process with continuous observation and deterministic sampling

Time average

If $X(t)$ a is continuous time process observed in $(0, T]$, then *time average* is defined by

$$\bar{X} = \frac{1}{T} \int_0^T X(t)\, \mathrm{d}t \tag{8.1}$$

In the case of a discrete space continuous time process, the state changes occur at discrete points in time. Hence, continuous observation can be realized by simply recording the times t_i of occurrence of a state change and the new state reached $X(t_i^+)$ (event-driven observation, see Figure 8.2 (a)). The time average can then be calculated as

$$\bar{X}_e = \frac{1}{T} \sum_{i=0}^{n-1} (t_{i+1} - t_i)\, X(t_i^+) + (T - t_n)\, X(t_n^+) \tag{8.2}$$

where n is the number of events in the interval $(0, T]$ and $t_0 = 0$.

Example 8.1 Interval availability
The estimation of a transient measure such as the interval availability in $(0, T]$ is done by accumulating the time the system is working during the interval $(0, T]$. Let $X(t)$ represent the state of the system at time t. $X(t)$ can only take two values: 0 (the system is failed) or 1 (the system is working). An estimate of the interval availability in over the r^{th} interval $(0, T]$ is:

$$\bar{A}^{(r)}(0, T] = \frac{1}{T} \sum_{i=0}^{n-1} \left(t_{i+1}^{(r)} - t_i^{(r)} \right) X(t_i^{(r)+})$$

In the case of a high dependability, note that the estimates $\bar{A}^{(r)}(0, T]$ do not follow a distribution which density function is symmetric around the mean (a large number of $\bar{A}^{(r)}(0, T]$ will be equal to 1). Accurate estimations may require the use of special techniques beyond the scope of this book.

When the probability that the process is in state x at an arbitrary point of time is equal to the relative portion of time the process is in state x (time average) the process is said to be *ergodic* [17].

Example 8.2 Estimation of state probabilities
Let $X(t)$ be a discrete space continuous time process. If the process is ergodic, the probability that the system is in state x is equal to the relative portion of time the process is in state x.
The time τ_x the process is in state x can be obtained by sampling the state of the process each time an event (state change) occurs (event-driven sampling, see for instance Figure 8.2 (a)).

$$\tau_x = \sum_{i=0}^{n-1} (t_{i+1} - t_i)\, I(X(t_i^+) = x) + (T - t_n)\, I(X(t_n^+) = x)$$

where n is the number of events in the interval $(0, T]$, t_i the time of occurrence of event i ($t_0 = 0$) and $X(t_i^+)$ the state reached at time t_i, and $I(A)$ denotes an indicator function. $I(A)$ is equal to 1 if A is true, to 0 otherwise.
The estimated probability that the process is in state x is the time average:

$$\bar{p}_x = \frac{\tau_x}{T}$$

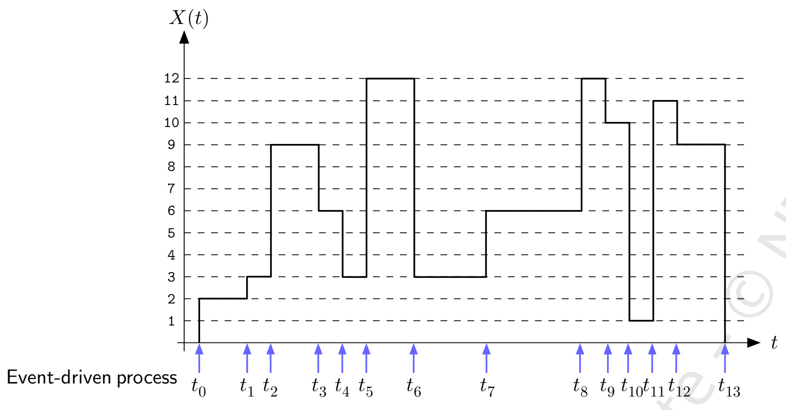
8.1.2 Sampling

Sampling converts a continuous time process into a discrete time process, i.e. $X_i = X(t_i)$. The sampling process may or may not depend on the observed process. Sampling the same process with a sampling process depending on the observed process and with a sampling process independent of the observed process generally leads to different results. For instance, the average residual work in a system where a new job is generated only after completion of the previous one is different depending on whether the state of the system is sampled:

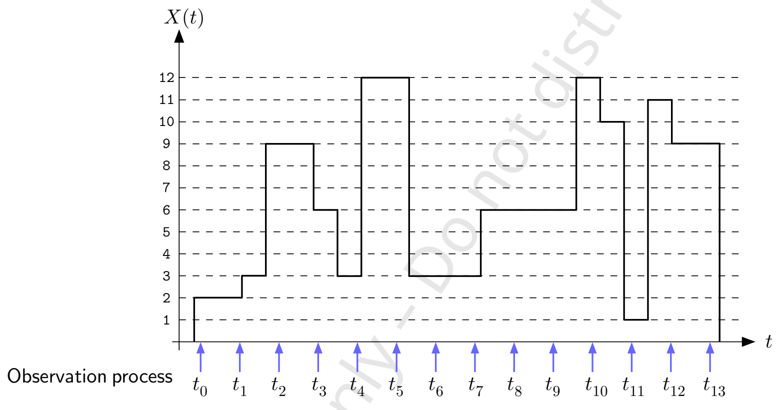
- independently, i.e. according to an *observation process*, or
- upon arrival of a new job, i.e. according to the *arrival process*.

Observation process

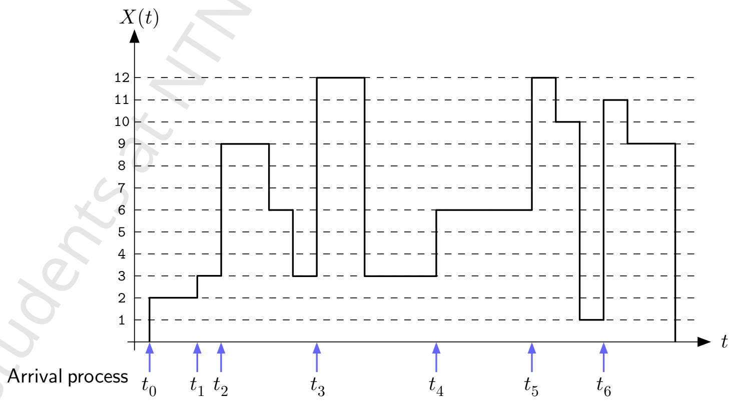
An observation process is an external process, independent of the observed process. It is typically a deterministic process, i.e. the observed process is sampled regularly (*equidistant sampling*) as in Figure 8.2 (b), but, more generally, any renewal process can be used, e.g. a Poisson process [28]. If the observed process is ergodic, an observation process can be used to sample the state of the observed process and estimate the probability that the process is in a state x eg a failed state, estimated by calculating the relative number of observations of the process in state x.



(a) At each state change (event-driven sampling)



(b) At regular times (equidistant sampling)



(c) At each arrival

Figure 8.2: Sampling of a process

Example 8.3 Estimation of state probabilities (Example 8.2 continued)
If $X(t)$ is an ergodic process, an alternative method to estimate the probability that the process is in state x is to use equidistant sampling. The probability that the process is in state x is then the ratio between the number of times n_x the process is observed in state x and the number of observations n.

$$\hat{p}_x = \frac{n_x}{n}, \text{ where } n_x = \sum_{i=1}^{n} I(X_i = x)$$

Arrival process

An arrival process is an example of sampling process depending on the observed process, see Figure 8.2 (c) for an illustration. Measurements based on an arrival process are meaningful only if the property studied depends on, or is influenced, by the arrivals. Examples include the call blocking probability, the waiting time distribution, the number of established connections when a link fails (the occurrence of a failure on the link is then the arrival), etc.

PASTA property

Let p_x denote the observer state distribution when the system is stationary and assume that the arrival process is a Poisson process with intensity λ. p_x is equal to the relative portion of time the system is in state x (time average). When the system is observed over a long period of time τ, the expected time the system is in state x is τp_x and the expected number of arrivals while the system is in state x is $\lambda\tau p_x$. The probability that an arrival finds the system in state x is equal to the number of arrivals while the system is in state x divided by the expected number of arrivals during τ.

$$P(\text{arrival sees the system in state } x) = \frac{\lambda\tau p_x}{\lambda\tau} = p_x \tag{8.3}$$

PASTA property (Poisson Arrivals See Time Averages)

For a stationary system with Poisson arrivals, the probability that an arrival finds the system in state x is equal to the probability that an outside observer finds the system in state x at an arbitrary point of time (time average, steady state probability).

In the case of a Poisson arrival process, and only in this case, observations of the arrival process can be used to estimate the system state probabilities. This is often a simpler an more effective method compared to sampling using an observation process.

Example 8.4 Difference between observation and arrival processes
In addition to event-driven sampling, Figure 8.2 illustrates two ways of sampling the process $X(t)$:

- equidistant sampling, i.e. at regular points in time independently of the observed process, and

- sampling at each arrival.

Sampling $X(t)$ using these two methods results in the following samples, respectively:

- equidistant sampling (14 observations):

$$\underline{\mathsf{X}}_o = \{2, 2, 9, 6, 3, 12, 3, 3, 6, 6, 12, 1, 9, 9\}$$

- sampling at each arrival (7 observations):

$$\underline{\mathsf{X}}_a = \{2, 3, 9, 12, 6, 12, 11\}$$

The corresponding sample means are, respectively:

$$\hat{X}_o = \frac{1}{14}\sum_{i=0}^{13} X_{oi} = \frac{1}{14}(2 + \ldots + 9) = 5.9286$$

$$\hat{X}_a = \frac{1}{7}\sum_{i=0}^{12} X_{ai} = \frac{1}{7}(2 + \ldots + 11) = 7.8571$$

$\hat{X}_o$ and $\hat{X}_a$ are different. It is therefore wrong in this case, where the arrival process is not a Poisson process, to use the arrival process to estimate the steady state probabilities.

8.2 Analysis of experimental data

In Section 8.1, we have seen how to observe a stochastic process and collect data. This section explains now how to make inferences from experimental data, i.e. how to estimate some parameters with a given level of confidence from a set of observations.

Formally, the set of all possible observations is called the *population*. An observation is a random variable and a set of observations is a *sample*. If the observations are independent and identically distributed the sample is a *random sample*. A function of the random variables constituting a random sample is called a *statistic*.

Random sample

A *random sample* is a set of independent and identically distributed observations.

8.2.1 Point estimation

Point estimate

A *point estimate* θ of a population parameter ξ is a value of a statistic Θ.

$$\theta = \Theta(X_1, X_2, \ldots, X_n) \tag{8.4}$$

where $\{X_1, X_2, \ldots, X_n\}$ is a sample of the population.
Θ is called a *point estimator*.

Before, we define standard point estimators such as the *sample mean*, we briefly list the desirable properties for a point estimator.

Desirable properties for a point estimator

Let Θ be a point estimator whose value θ is a point estimate of a parameter ξ. Desirable properties for a point estimator are:

- *Unbiasedness.* A point estimator Θ is an *unbiased estimator* of a parameter ξ if and only if $\mathrm{E}(\Theta) = \xi$.
- *Efficiency.* The estimator should also have the smallest possible variance. The unbiased estimator of a parameter ξ with smallest variance is called the most efficient estimator of ξ.
- *Consistency.* The estimator should converge to the parameter estimated when the sample size n grows to infinity. $\Theta(n)$ is a *consistent estimator* of a parameter ξ if $\lim_{n\to\infty} \mathrm{Var}(\Theta(n)) = 0$.

Example 8.5 Estimation of state probabilities (Example 8.3 continued)
If $X(t)$ is an ergodic process, $\hat{p}_x = \frac{n_x}{n}$, where $n_x = \sum_{i=1}^{n} I(X_i = x)$ is an unbiased and consistent estimator of the probability that the process is in state x.

$$\mathrm{E}(\hat{p}_x) = \mathrm{E}\Big(\frac{n_x}{N}\Big) = \frac{1}{N}\,\mathrm{E}\left(\sum_{i=1}^{N} I(X_i = x)\right) = \frac{1}{N}\sum_{i=1}^{N}\mathrm{E}(I(X_i = x)) = \frac{1}{N} N p_x = p_x$$

$$\lim_{n\to\infty} \hat{p}_x = p_x$$

Point estimators

Let us consider a population of expected value μ and variance σ^2.

Sample mean

The *sample mean* $\hat{X}$ of a random sample $X = \{X_1, X_2, \ldots, X_n\}$ is defined by

$$\hat{X} = \frac{1}{n}\sum_{i=1}^{n} X_i \tag{8.5}$$

$$\mathrm{E}(\hat{X}) = \frac{1}{n}\sum_{i=1}^{n}\mathrm{E}(X_i) = \frac{1}{n} n\mu = \mu \tag{8.6}$$

Hence, the sample mean $\hat{X}$ is an unbiased estimator of the expected value μ of the population.

Besides, since the X_i's are independent, using (2.68), we have:

$$\mathrm{Var}(\hat{X}) = \frac{1}{n^2}\sum_{i=1}^{n}\mathrm{Var}(X_i) = \frac{1}{n^2}n\sigma^2 = \frac{\sigma^2}{n} \tag{8.7}$$

and therefore

$$\lim_{n\to\infty}\mathrm{Var}(\hat{X}) = \lim_{n\to\infty}\frac{\sigma^2}{n} = 0 \tag{8.8}$$

that is, the sample mean $\hat{X}$ is also a consistent estimator of the expected value μ of the population.

Note that, for a finite number of observations, $n < \infty$, $\mathrm{Var}(\hat{X}) > 0$. In other words, there is an uncertainty on the estimate. How to quantify this uncertainty is discussed in Section 8.2.2. Note also that the variance of $\hat{X}$ is different from (is smaller than) the variance of the stochastic process itself, $\mathrm{Var}(\hat{X}) < \mathrm{Var}(X(t)),\ n > 1$.

Sample variance

The *sample variance* S^2 of a random sample $X = \{X_1, X_2, \dots, X_n\}$ is defined by

$$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \hat{X})^2 = \frac{1}{n-1}\sum_{i=1}^{n}X_i^2 - \frac{n}{n-1}\hat{X}^2 \tag{8.9}$$

S is the *sample standard deviation.*

$$\mathrm{E}(S^2) = \frac{1}{n-1}\,\mathrm{E}\left(\sum_{i=1}^{n}(X_i - \hat{X})^2\right) \tag{8.10}$$

Using that

$$\sum_{i=1}^{n}\Big((X_i-\mu)^2 - (\hat{X}-\mu)^2\Big) = \sum_{i=1}^{n}X_i^2 - n\hat{X} = \sum_{i=1}^{n}\Big(X_i - \hat{X}\Big)^2 \tag{8.11}$$

Eq. (8.10) becomes

$$\begin{aligned}\mathrm{E}(S^2) &= \frac{1}{n-1}\left(\sum_{i=1}^{n}\mathrm{E}\big((X_i-\mu)^2\big) - \sum_{i=1}^{n}\mathrm{E}\Big((\hat{X}-\mu)^2\Big)\right)\\ &= \frac{1}{n-1}\Big(n\sigma^2 - n\mathrm{Var}(\hat{X})\Big)\end{aligned} \tag{8.12}$$

and replacing $\mathrm{Var}(\hat{X})$ by the expression obtained in (8.7), we have

$$\mathrm{E}(S^2) = \frac{1}{n-1}\left(n\sigma^2 - \sigma^2\right) = \sigma^2 \tag{8.13}$$

The sample variance is therefore an unbiased estimator of the variance σ^2 of the population It can also be shown that the sample variance is consistent estimator of σ^2, see for example [19].

Variance of the sample mean

$$S_{\hat{X}}^2 = \frac{S^2}{n} \tag{8.14}$$

$S_{\hat{X}}^2$ is an unbiased and consistent estimator of the variance of the sample mean $\hat{X}$.

Standard error of the sample mean

$$S_{\hat{X}} = \frac{S}{\sqrt{n}} \tag{8.15}$$

$S_{\hat{X}}$ is a consistent estimator of the standard error, or standard deviation, of the sample mean $\hat{X}$, but note that it is biased since the expected value of the square root of a random variable is not equal to the square root of the expected value, $\mathrm{E}(\sqrt{X}) \neq \sqrt{\mathrm{E}(X)}$, [18].

8.2.2 Interval estimation

Interval estimate

An *interval estimate* of a population parameter ξ is an interval, or *interval estimator*, $\theta_L \leqslant \xi < \theta_U$ where θ_L and θ_U depend on the value of a statistic Θ for a particular random sample and on the distribution of the statistic (*sampling distribution*).

θ_L and θ_U depend on the sample. Therefore, they are values of random variables Θ_L and Θ_U, respectively. From the sampling distribution of Θ it is possible to find Θ_L and Θ_U such that

$$P(\Theta_L \leqslant \xi < \Theta_U) = 1 - \alpha,\ 0 < \alpha < 1 \tag{8.16}$$

The interval $[\Theta_L, \Theta_U)$ is called the *confidence interval* and $(1 - \alpha)$ the *confidence coefficient* or *degree of confidence*. The confidence interval contains the value ξ with probability $1 - \alpha$.

In the following, we detail how to establish a confidence interval in the case of a sampling distribution which density function is symmetric around the mean. How to establish a confidence interval in the case of a non-symmetric sampling distribution is beyond the scope of this book.

Quantiles

Let Z be a random variable of continuous cumulative distribution function $F_Z(z)$.

$(1-\alpha)$-quantile

z_α is the $(1-\alpha)$-quantile of $F_Z(z)$ if and only if

$$F_Z(z_\alpha) = P(Z \leqslant z_\alpha) = 1 - \alpha \tag{8.17}$$

Rearranging (8.17) yields

$$P(Z > z_\alpha) = 1 - F_Z(z_\alpha) = \alpha \tag{8.18}$$

When the probability density function $f_Z(z)$ of Z is even, i.e. $f_Z(z) = f_Z(-z)$,

$$P(Z \leqslant -z_{\alpha/2}) = P(Z > z_{\alpha/2}) = \frac{\alpha}{2} \tag{8.19}$$

which gives

$$P(-z_{\alpha/2} < Z \leqslant z_{\alpha/2}) = 1 - \alpha \tag{8.20}$$

This is illustrated in Figure 8.3.

Examples of distributions with an even probability density function include the standard normal distribution, $\mathcal{N}(0,1)$, and the Student's t-distribution.

Central limit theorem

Central limit theorem

If $X = \{X_1, X_2, \ldots, X_n\}$ is a random sample of size n taken from a population with expected value $\mathrm{E}(X_i) = \mu$ and variance $\mathrm{Var}(X_i) = \sigma^2 < \infty, \;\; i = 1, 2, \ldots, n$, then the sample mean $\hat{X}$ asymptotically follows a normal distribution with expected value μ and variance σ^2/n as $n \to \infty$.

$$\hat{X} \underset{n\to\infty}{\sim} \mathcal{N}\left(\mu, \frac{\sigma^2}{n}\right) \tag{8.21}$$

That is, if the random variable Z is defined as

$$Z = \frac{\hat{X} - \mu}{\sigma/\sqrt{n}} \tag{8.22}$$

it asymptotically follows a standard normal distribution as $n \to \infty$, $Z \underset{n\to\infty}{\sim} \mathcal{N}(0,1)$.

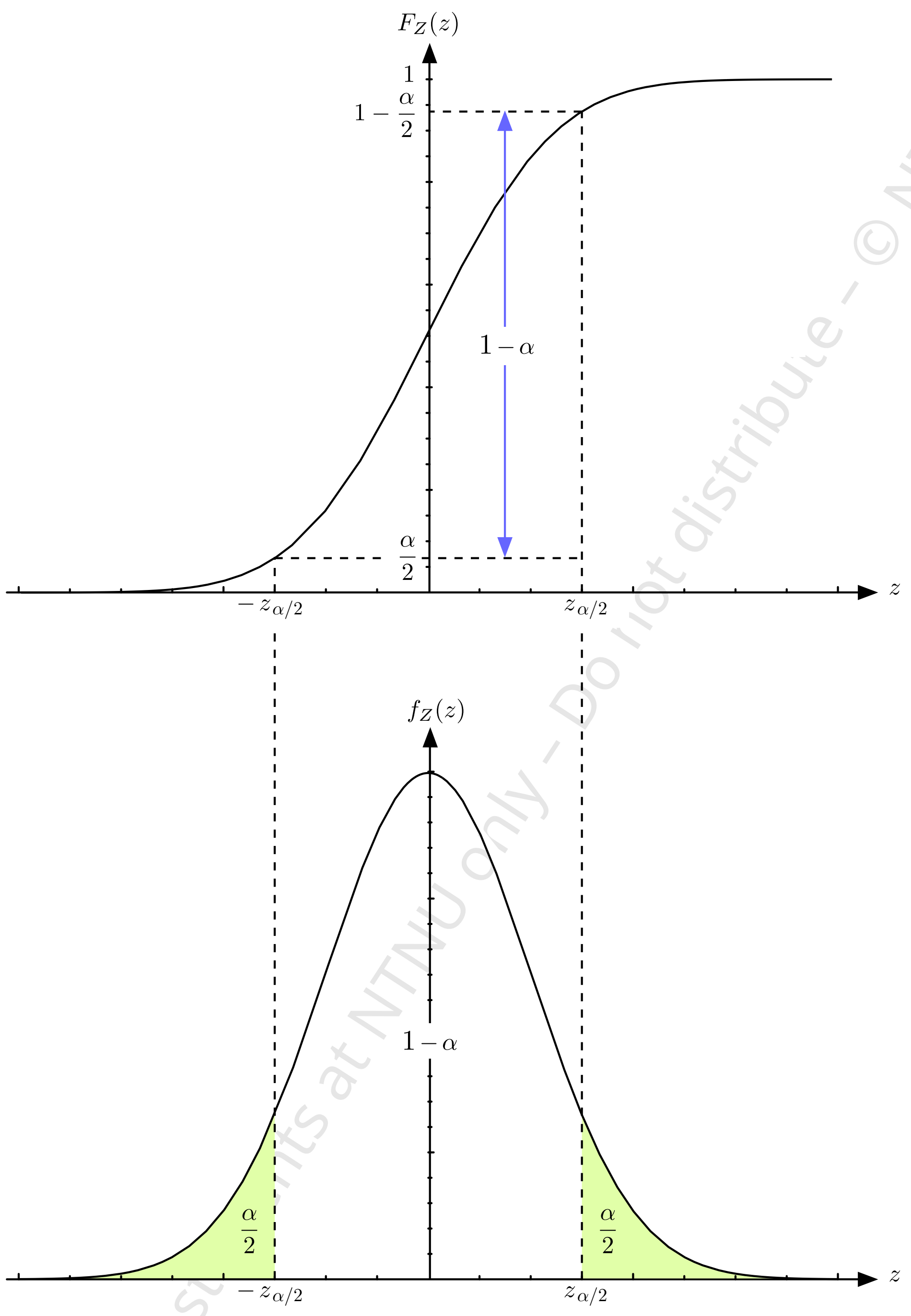


Figure 8.3: Even density function and quantiles

Interval estimation of the expected value with known variance

Using (8.20), we can then write

$$\begin{aligned} & P\left(-z_{\alpha/2} < \frac{\hat{X}-\mu}{\sigma/\sqrt{n}} \leqslant z_{\alpha/2}\right) = 1-\alpha \\ \Leftrightarrow \quad & P\left(\hat{X} - z_{\alpha/2}\frac{\sigma}{\sqrt{n}} \leqslant \mu < \hat{X} + z_{\alpha/2}\frac{\sigma}{\sqrt{n}}\right) = 1-\alpha \end{aligned} \tag{8.23}$$

Confidence interval of μ (known variance)

If $X = \{X_1, X_2, \ldots, X_n\}$ is a random sample of size n taken from a population with expected value $\mathrm{E}(X_i) = \mu$ and known variance $\mathrm{Var}(X_i) = \sigma^2 < \infty,\ \ i = 1, 2, \ldots, n$, and $\hat{X}$ is the sample mean, then a $(1-\alpha)$-*confidence interval* for μ is given by

$$\hat{X} - z_{\alpha/2}\frac{\sigma}{\sqrt{n}} \leqslant \mu < \hat{X} + z_{\alpha/2}\frac{\sigma}{\sqrt{n}} \tag{8.24}$$

where $z_{\alpha/2}$ is the $(1-\alpha/2)$-quantile of the standard normal distribution.

In other words, the probability that the error in estimating μ by $\hat{X}$ exceeds $\pm z_{\alpha/2}\frac{\sigma}{\sqrt{n}}$ is equal to α. Note that the width of the interval estimate shrinks with $\sqrt{n}$ as n increases.

The other way around, the sample size n can be chosen such that the error does not exceed a given value e with a probability $(1-\alpha)$.

$$n \geqslant \left(\frac{z_{\alpha/2}\sigma}{e}\right)^2 \implies P\left(e = \left|\mu - \hat{X}\right| \leqslant z_{\alpha/2}\frac{\sigma}{\sqrt{n}}\right) \leqslant 1-\alpha \tag{8.25}$$

Interval estimation of the expected value with unknown variance

When the variance of the population is unknown, the sample variance S^2 is used instead to define the confidence interval of μ.

Let

$$T = \frac{\hat{X}-\mu}{S/\sqrt{n}} \tag{8.26}$$

T follows a Student's t-distribution with $n-1$ degrees of freedom.

Confidence interval of μ (unknown variance)

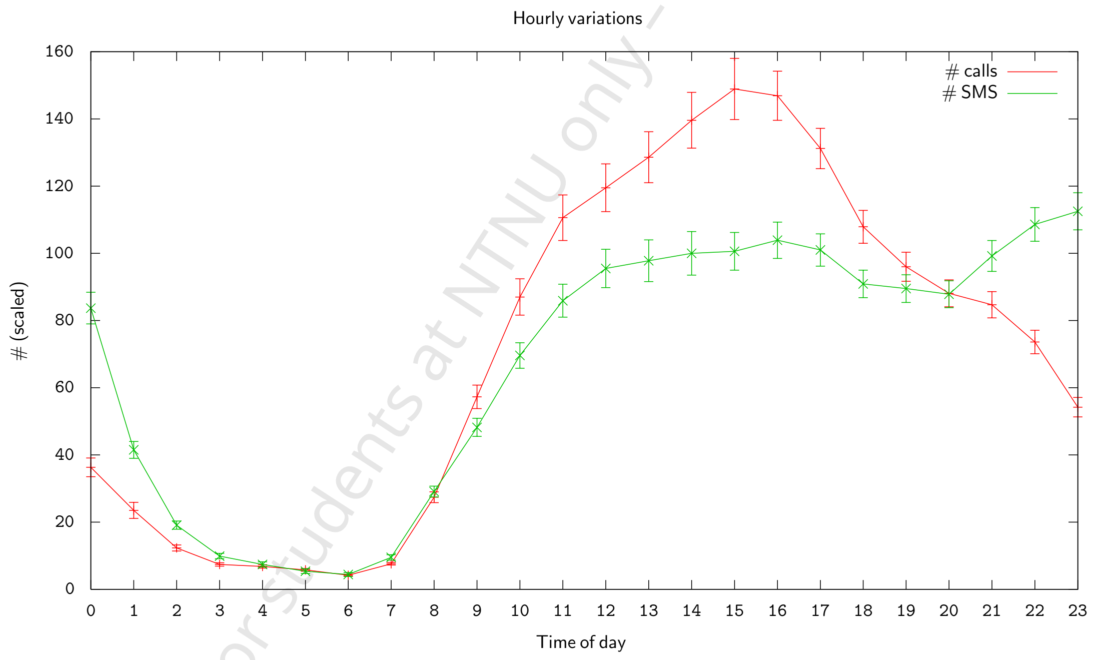
If $X = \{X_1, X_2, \dots, X_n\}$ is a random sample of size n taken from a population with expected value $\mathrm{E}(X_i) = \mu$ and unknown variance $\mathrm{Var}(X_i) = \sigma^2 < \infty$, $i = 1, 2, \dots, n$, and $\hat{X}$ is the sample mean, then a $(1-\alpha)$-confidence interval for μ is given by

$$\hat{X} - t_{\alpha/2,n-1}\frac{S}{\sqrt{n}} \leqslant \mu < \hat{X} + t_{\alpha/2,n-1}\frac{S}{\sqrt{n}} \qquad (8.27)$$

where $t_{\alpha/2,n-1}$ is the $(1-\alpha/2)$-quantile of the Student's t-distribution with $n-1$ degrees of freedom.

For large values of n, the Student's t-distribution can be approximated by the standard normal distribution. As a rule of thumb, when $n \geqslant 30$ the Student's t-distribution can be replaced by the standard normal distribution.

Example 8.6 Hourly variations of the traffic load on a base station
A measurement study of the traffic load on a mobile phone base station has been carried out on 73 base stations over one week. The number of calls and the number of short text messages (SMSs) handled per hour are recorded each hour on each base station. The observations are assumed to be independent and identically distributed. The sample mean and the standard error of sample mean are plotted in the figure below. Note that the number of calls and the number of SMS are scaled with an unknown factor different for each base station. The absolute numbers are therefore incorrect, but the ratio between number of calls and SMSs is correct.



8.2.3 Effect of autocorrelation

We have so far only considered random samples, i.e. we have assumed the observations X_i to be independent and identically distributed. In the following, we relax the assumption of independence and discuss the consequences on the estimation of population parameters, in particular the sample mean and the sample variance.

Let $X = \{X_1, X_2, \ldots, X_n\}$ be a sample of identically distributed observations with possible dependence between the X_i's, and $\mathrm{E}(X_i) = \mu$ and $\mathrm{Var}(X_i) = \sigma^2,\ i = 1, 2, \ldots, n$.

Sample mean

The expected value of a sum is always equal to the sum of expected values. Eq (8.6) is therefore unchanged, and the sample mean remains an unbiased estimator of the expected value of the population.

The dependence affects however the variance of the estimate since the variance of the sum is equal to the sum of the variance only in the case of uncorrelated random variables, see page 76. In the general case of non-independent variables, the variance of a sum of random variables is given by (2.67) and therefore the variance of $\hat{X}$ is:

$$\mathrm{Var}(\hat{X}) = \mathrm{Var}\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_{X_iX_j} \tag{8.28}$$

where $\sigma_{X_iX_j}$ is the autocovariance between X_i and X_j.

If the observed process is covariance stationary, the autocovariance is finite and only depends on the difference $j - i$, $\sigma_{X_iX_j} = \sigma_{X_{i+k}X_{j+k}} \equiv \sigma_{XX}(j-i) < \infty,\ \forall k \in \mathbb{Z}$, see page 99. Hence, as all terms with the same $(j - i)$ in (8.28) are equal, (8.28) can be simplified to:

$$\mathrm{Var}(\hat{X}) = \frac{1}{n^2}\sum_{i=1-n}^{n-1}(n - |i|)\sigma_{XX}(i) \tag{8.29}$$

Introducing the autocorrelation $\rho_X(i) = \dfrac{\sigma_{XX}(i)}{\sigma^2}$ (3.25)

$$\mathrm{Var}(\hat{X}) = \frac{\sigma^2}{n}\sum_{i=-(n-1)}^{n-1}\left(1 - \frac{|i|}{n}\right)\rho_X(i) \tag{8.30}$$

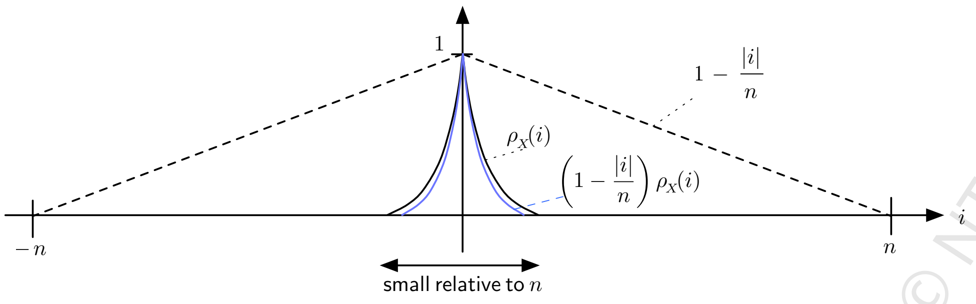


Figure 8.4: $\left(1-\frac{|i|}{n}\right)\rho_X(i) \approx \rho_X(i),\ \forall i$ **for large** n

Finally, since the autocorrelation is an even function,

Variance of $\hat{X}$

$$\text{Var}(\hat{X}) = \frac{\sigma^2}{n} + \frac{2\sigma^2}{n}\sum_{i=1}^{n-1}\left(1-\frac{i}{n}\right)\rho_X(i) \tag{8.31}$$

The first term corresponds to the variance of $\hat{X}$ in the case of uncorrelated observations (8.7). The second term is the contribution from the dependence between the X_i's. If consecutive observations are positively correlated, $\rho_X(i) > 0$, $\text{Var}(\hat{X})$ is larger than in the case of uncorrelated observations. If consecutive observation are negatively correlated, $\rho_X(i) < 0$, $\text{Var}(\hat{X})$ is smaller than in the case of uncorrelated observations.

Now if $\lim_{i\to\infty}\rho_X(i) = 0$ and $\lim_{n\to\infty}\rho_X(i)\frac{i}{n} = 0,\ \forall i,$

$$\lim_{n\to\infty}\sum_{i=1}^{n-1}\left(1-\frac{i}{n}\right)\rho_X(i) = \sum_{i=1}^{\infty}\rho_X(i) \tag{8.32}$$

Hence, under the assumption that $\lim_{i\to\infty}\rho_X(i) = 0$, we can write the following approximation for large n (n is "large enough" if $\frac{i}{n}\rho_X(i) \approx 0,\ \forall i$, see Figure 8.4 for an illustration)

$$\text{Var}(\hat{X}) \approx \frac{\sigma^2}{n}\left(1 + 2\sum_{i=1}^{\infty}\rho_X(i)\right) = \frac{\sigma^2}{n}(1+c) = \frac{\sigma^2}{m}$$
$$\text{where } c = 2\sum_{i=1}^{\infty}\rho_X(i) \text{ and } m = \frac{n}{1+c} = \frac{n}{\sum_{i=-\infty}^{\infty}\rho_X(i)} \tag{8.33}$$

m is the *equivalent sample size*, i.e. the number of independent observations needed to obtain the same variance. $m = n$ when the observations are uncorrelated, $m > n$ when the observations are positively correlated, and $m < n$ when the observations are negatively correlated.

If c is large and the cost (in terms of computer time for a simulation experiment or work for a measurement study on a real system) is large, it may be relevant to make observations less often to reduce c and the overall cost.

Sample variance

In the case of non-independent observations, the sample variance is no longer an unbiased estimator of the variance of the population.

From (8.10)

$$\mathrm{E}(S^2) = \frac{1}{n-1}\left(n\sigma^2 - n\mathrm{Var}(\hat{X})\right) \tag{8.34}$$

Replacing $\mathrm{Var}(\hat{X})$ by the expression obtained in (8.33),

Expected value of the sample variance S^2

$$\mathrm{E}(S^2) = \frac{1}{n-1}\left(n\sigma^2 - (1+c)\,\sigma^2\right) = \sigma^2\left(1 - \frac{c}{n-1}\right) \tag{8.35}$$

- if $c = 0$ (uncorrelated observations) then $\mathrm{E}(S^2) = \sigma^2$ and the sample variance is thus an unbiased estimator of the variance of the population.
- if $c > 0$ (globally positively correlated observations) then $\mathrm{E}(S^2) < \sigma^2$ and the sample variance is thus a biased estimator of the variance of the population. If it is used to establish a confidence interval, see Section 8.2.2, it leads to a tight but unsafe interval.
- if $c < 0$ (globally negatively correlated observations) then $\mathrm{E}(S^2) > \sigma^2$and the sample variance is thus a biased estimator of the variance of the population. If it is used to establish a confidence interval, the interval will be too loose.

8.3 Experiment setup

In the previous sections, we have seen how to observe a process and how to analyze experimental data and estimate parameters in the case of independent and identically distributed observations and determine confidence interval in the case of sampling distributions which density function is symmetric around the mean. Section 8.3.1 discusses what to record to be able to derive estimates and how to possibly reduce the amount of data to store. Next, when the stochastic process observed does not naturally results in independent observations, the experiment must be organized in order to achieve independence between the observations. Furthermore, to derive interval estimates using the expressions given in Section 8.2.2, Two methods to get independent and identically normally distributed observations are presented in Section 8.3.2, namely the replication method and the sectioning method.

8.3.1 Accumulating statistics

Sampling or continuous event-driven observation may produce a large amount of data, and, hence, demand large processing and storage capacities unless it is possible to reduced the data collected. Measurements at the packet level in the Internet are a typical example where measurements even over a limited time period may result in gigabytes of data that need to be parsed and stored. There has been a lot of work on methods to collect data in the Internet, see for instance `http://www.nlanr.net`. However, if the objectives of the measurement study and the relevant estimators are well defined, the amount of data can be considerably reduced by accumulating statistics on the fly. For instance, if one is interested only in the sample mean and the sample variance, only the number of observations and the sum and the quadratic sum of the observations need to be recorded, see (8.5) and (8.9). They are then called *sufficient observators*. The drawback is that the measurement data cannot be used to derive an estimate that has not been accumulated during the experiment and the measurement study must be repeated.

As an example if one is only interested in the average number of packets in a buffer, the number of packets observed can be accumulated during the experiment and divided by the number of observations at the end of the experiment. Storing the sum dramatically reduces the storage requirements compared to storing each observation. But, the sum can only be used to compute the sample mean of the number of packets in the buffer and not other metrics such as the packet inter-arrival time, the distribution of the number of packets in the buffer etc.

8.3.2 Achieving independence between the observations

Replication method

A method to obtain n independent and identically distributed observations consists in replicating (*replication* method) the same experiment n times with the same initial and terminating conditions, see Figure 8.5. In a simulation experiment, different seeds are used for the random number generator to ensure different stochastic evolutions. This method has been developed for transient state analysis of a system, but is also in principle applicable to the study of steady state properties. In a simulation study, this method allows to run several simulation jobs (replicates) in parallel on different computers.

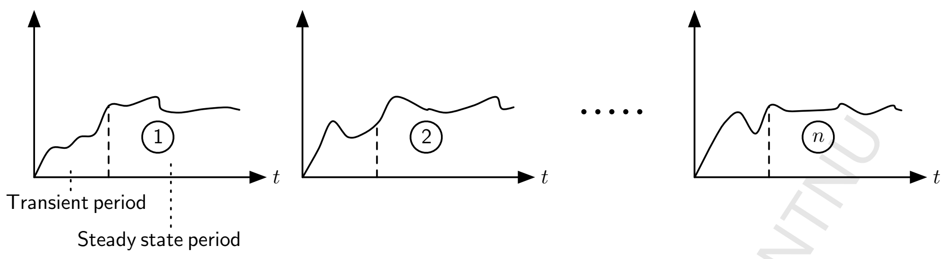


Figure 8.5: Independence through replication

The n replications of the same experiment with identical initial and terminating conditions yields n samples for which the sample mean can be calculated. In the following, the terminating condition is a given number m_x of samples.

$$
\begin{aligned}
&\text{Repl. 1: } \underline{X}_1 = \{X_{11}, X_{12}, \ldots, X_{1m_1}\} \implies \hat{X}_1 = \frac{1}{m_1}\sum_{i=1}^{m_1} \hat{X}_{1i} \\
&\text{Repl. 2: } \underline{X}_2 = \{X_{21}, X_{22}, \ldots, X_{2m_2}\} \implies \hat{X}_2 = \frac{1}{m_2}\sum_{i=1}^{m_2} \hat{X}_{2i} \\
&\ldots \\
&\text{Repl. } n\text{: } \underline{X}_n = \{X_{n1}, X_{n2}, \ldots, X_{nm_n}\} \implies \hat{X}_n = \frac{1}{m_n}\sum_{i=1}^{m_n} \hat{X}_{ni}
\end{aligned}
\tag{8.36}
$$

Note that the terminating condition can also be a given time duration and then the number of observations for each replicate may differ.

The sample means constitute a new sample $\underline{X} = \{\hat{X}_1, \hat{X}_2, \ldots, \hat{X}_n\}$, where the $\hat{X}_i$'s are independent and identically distributed (random sample, same parameters and same initial and terminating conditions). Therefore, the sample mean and the sample variance can be calculated

$$
\begin{aligned}
\hat{X}_R &= \frac{1}{n}\sum_{r=1}^{n} \hat{X}_r \\
S_R^2 &= \frac{1}{n-1}\sum_{r=1}^{n} (\hat{X}_r - \hat{X}_R)^2
\end{aligned}
\tag{8.37}
$$

and the $(1-\alpha)$-confidence interval is

$$
\left[\hat{X}_R - t_{\alpha/2,n-1}\frac{S_R}{\sqrt{n}}, \hat{X}_R + t_{\alpha/2,n-1}\frac{S_R}{\sqrt{n}}\right)
\tag{8.38}
$$

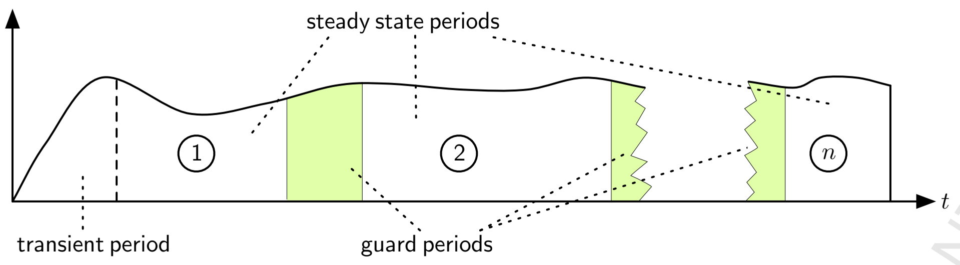


Figure 8.6: Independence through sectioning

Sectioning (batch means) method

When the replication method is used for the study of steady state properties, it incurs a large overhead since n transient periods must be thrown away, see Figure 8.5.

To avoid this drawback, an alternative approach, denoted *sectioning* or *batch-means*, is to run a single experiment and divide the observation period in n independent sections or batches. To ensure that the observations can be assumed independent, a guard period, typically significantly shorter than the transient period, may be introduced between the observation periods, see Figure 8.6. Beyond a distance L between two observations, the correlation between the observations becomes negligible. The guard period should be at least equal to L. If the length of the observation period is very large with respect to L, the correlation between estimates of consecutive observation periods is negligible and no guard period is needed.

An experiment yields one sample of m observations $\underline{X} = \{X_1, X_2, \ldots, X_m\}$, where the X_i's are neither normally distributed nor independent. The sectioning method consists in partitioning the sample into n sub-samples ($n < m$) and calculating the sample mean for each sub-sample.

$$\begin{aligned}
&\text{Batch 1: } \underline{X}_1 = \{X_1, X_2, \ldots, X_{m/n}\} \implies \hat{X}_1 = \frac{n}{m}\sum_{i=1}^{m/n}\hat{X}_i \\
&\text{Batch 2: } \underline{X}_2 = \{X_{1+m/n}, X_{2+m/n}, \ldots, X_{2m/n}\} \implies \hat{X}_2 = \frac{n}{m}\sum_{i=1+m/n}^{2m/n}\hat{X}_i \\
&\ldots \\
&\text{Batch } n\text{: } \underline{X}_n = \{X_{1+(n-1)m/n}, \ldots, X_m\} \implies \hat{X}_n = \frac{n}{m}\sum_{i=1+(n-1)m/n}^{m}\hat{X}_i
\end{aligned} \tag{8.39}$$

The sample means constitute a new sample $\underline{X} = \{\hat{X}_1, \hat{X}_2, \ldots, \hat{X}_n\}$, where the $\hat{X}_i$'s are independent and identically distributed. If n/m is sufficiently large, the $\hat{X}_i$'s follow asymptotically a normal distribution (8.21). Therefore, the sample mean and the sample variance can be calculated

$$\begin{aligned}\hat{X}_B &= \frac{1}{n}\sum_{b=1}^{n}\hat{X}_b \\ S_B^2 &= \frac{1}{n-1}\sum_{b=1}^{n}(\hat{X}_b - \hat{X}_B)^2\end{aligned} \tag{8.40}$$

and the $(1-\alpha)$-confidence interval is

$$\left[\hat{X}_B - t_{\alpha/2,n-1}\frac{S_B}{\sqrt{n}}, \hat{X}_B + t_{\alpha/2,n-1}\frac{S_B}{\sqrt{n}}\right) \tag{8.41}$$

There is a trade-off between the number n of batches and the number m/n of observations per batch. On one hand, a large n is desirable so that the confidence interval is as tight as possible. But the larger the number of batches, the fewer the number of observations per batch and, on the other hand, m/n must be as large as possible so that the $\hat{X}_i$'s can be assumed to be normally distributed.[1]

As a rule of thumb, authors in [18] advise to choose n between 12 and 20 when $m > 1000$ and to prefer a larger number of observations per batch rather than a number n of batches larger than 20.

[1]A large number of observations per batch is also desirable when asymptotically unbiased estimators are used.

9 Summary and concluding remarks

So, at the end, what is the main content of this book? Figure 9.1 illustrates how the basic theory of probability, statistics and stochastic processes is used as a fundament for dealing with the dependability, the performance and the QoS delivered in various application contexts. The objective of this book is to give an understanding of how to approach these issues in a quantitative manner and to give a firm basis on how to deal with them. This book is introductory, and the use and the further development of ICT pose tremendous challenges to the aspects treated. Hence, supplementary insight is likely to be needed. It is the intent that this book should enable the reader to acquire additional competences in both the more theoretical and applied domains. Using Figure 9.1 as a guide, the following briefly summarizes the more salient issues covered by this book.

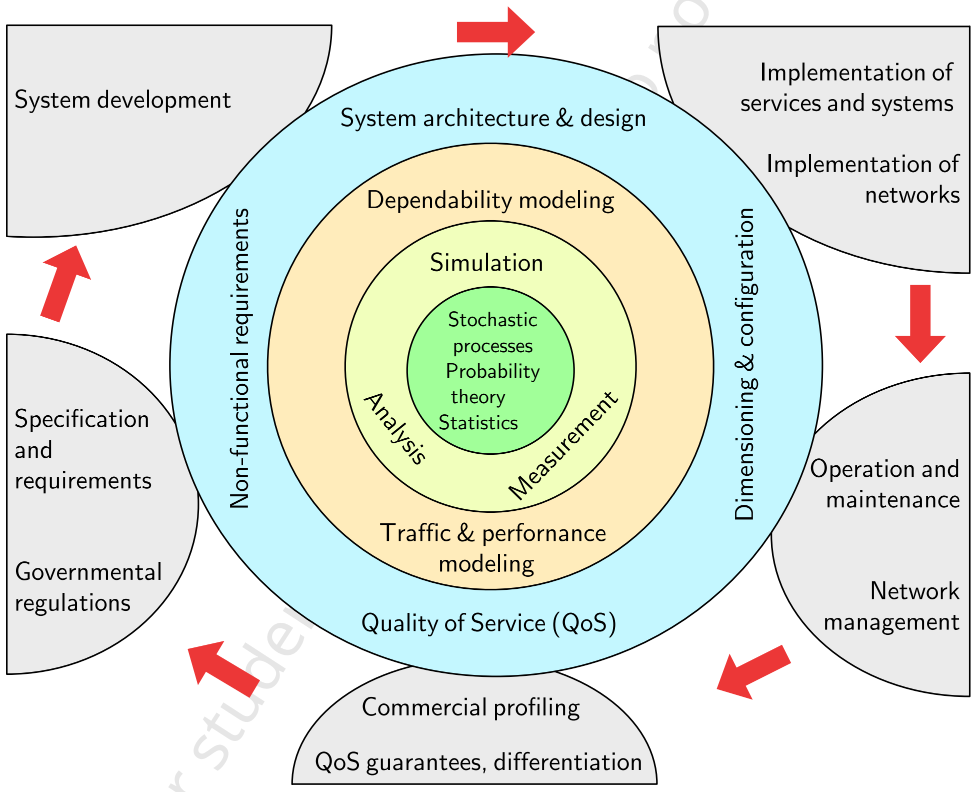

System development
Implementation of services and systems
Implementation of networks
System architecture & design
Dependability modeling
Simulation
Stochastic processes
Probability theory
Statistics
Analysis
Measurement
Non-functional requirements
Dimensioning & configuration
Traffic & performance modeling
Quality of Service (QoS)
Specification and requirements
Governmental regulations
Operation and maintenance
Network management
Commercial profiling
QoS guarantees, differentiation

Figure 9.1: Summary of this book

Starting from the core; to be able to deal with real life systems, a true understanding of randomness and stochastic behavior is necessary. The workload put on systems as well as their operational status with respect to failure and repair are governed by stochastic processes. Hence, the behavior of systems is stochastic and concepts like transience and (quasi) stationarity are important. Furthermore, to be able to extract the underlying behavior of systems, to make statistical inferences and to control the uncertainties of the estimates, are necessary skills for engineering ICT-systems. For these reasons, Chapter 2 refreshes probability theory and statistics in an applied context, and Chapter 3 gives an introduction to and an overview of the area of stochastic processes. Moving outwards to the next level in Figure 9.1, this core knowledge is used for simulation, analysis and measurements.

Since systems usually are too complex to allow a precise mathematical analysis, *simulation* is a useful tool to investigate their properties and to get a thorough insight into their operation. There are "drag and drop" simulation packages on the market which make it easy to set up a simulation scenario by using components of a package. In some contexts these may be very useful. In others, for instance when developing a new functionality, problems cannot be solved using a "drag and drop" approach. Anyhow, understanding what is going on is necessary to be able to use simulation properly. For this reason, the basic building blocks and methodologies of simulation are introduced in Chapter 4. In simulation modeling, as in other modeling approaches, the challenge is to identify the important features of the system and transform these into a model. To get a full grasp of this, it is recommended that the reader familiarizes herself/himself with a simulation language/tool that focuses on resource handling and synchronization primitives. For instance, Simula/DEMOS [24, 5], in spite of its old-fashioned interface and limited development environment, conveys this insight in an excellent manner.

In Chapters 5, 6 and 7, the objective has been to train the reader's probabilistic thinking about systems using *analysis*. Continuous time discrete state Markov models and analysis have been introduced as a tool. This class of models represents a compromise with a reasonable trade-off between model accuracy and mathematical and numerical tractability. For this reason, necessary assumptions must be made and care should be taken not to exceed the range of validity of the models. This also applies for the combinatorial models and analysis introduced. Some readers may find the mathematical analysis cumbersome. However, the analysis applied in this book is mathematically not very advanced and rather procedural, and it is acquired with some training. Furthermore, the system models used for introduction and illustration are simple, but once the principles and procedures are understood, they may be applied to more complicated/realistic systems.

Markovian models can be made quite complex using a big state space and matrix formulations. To deal with the complexity they are often defined by formalisms where the state space is not explicitly defined, for instance by various variants of stochastic Petri nets (SPN). Markovian models can also be combined with other types of models (e.g. fluid flow, imbedded Markov chain). However, also other types of models are widely applied today as for instance Gaussian models. New types of models are steadily seen as new systems and their environments require it. Still, simple Markovian models as presented in this book give a basic understanding of both system behavior and modeling techniques.

The dealings with *measurements* in Chapter 8 are rather brief and are concentrated on conveying two main messages:

- Be in control of what you are measuring and have a proper understanding of stationarity, bias, etc.
- Find the accuracy of what you observe. A consistent dealing with the accuracy of measurements, in all the phases from planning to results presentation, is the main difference between those who can and those who cannot deal with measurements in a professional and scientific appropriate manner.

Moving further outwards in Figure 9.1, we come closer to the real word and and meet what the authors regard as the main challenge in dealing with dependability and performance of systems, *modeling*. In modeling, the first step is to get an understanding of what are the properties of interest. The next step is to extract the system features of importance with respect to these properties and include them into a model. This requires a good insight into and understanding of the modeled system as well as command of the modeling tool(s). This must be trained! (It is a bit like riding a bicycle. It looks simple and the principles are simple, as the models presented in this book may look simple. However, when mounting a bike for the first time, it is not that straightforward, but with training... A recommendation to those using this book as a course textbook: do not let the exam be your first ride/independent modeling effort!) Two more issues related to modeling are:

- A model is made with an objective. The modeling details and the tool chosen may be highly dependent on this objective. For instance, if the objective is a better understanding of the overall behavior of a system this may lead to a model that is different from a model which has as its objective to get a detailed operational assessment of the properties of the system.
- There are some standard models and corresponding formulas, especially in traffic theory. These are often useful, but not applicable to all systems. Do not be tempted to transform all problems into nails because you have a hammer.

Moving further outward in Figure 9.1, it is illustrated how modeling and assessment of properties integrate with other system engineering and operation activities like system design and architecture, dimensioning and configuration, as well as monitoring and provisioning of a controlled QoS to the users of a system. The ability of an ICT-system to handle failures and the traffic load is commonly what makes a system “good or bad”. QoS is as important as the functionality itself. If the service is too poor, the system is useless. If a proper QoS (not too good) is not provided in a cost-efficient manner, the system is likely to be too costly, and to encounter difficulties in the highly competitive

ICT marketplace. As illustrated along the rim of Figure 9.1, what has been introduced in this book has a wide range of applications. These are highly dynamic and will change as the technology evolves. Hopefully, at least this is the intention, what is presented in this book gives the fundament for dealing with dependability and performance issues in current and future systems. Before putting the book away, the reader is encouraged to revisit Chapter 1 which provides an understanding of some of the application aspects as well as an overview of the concepts summarized in Figure 9.1.

References

[1] A. Høyland, *Sannsynlighetsregning og statistisk metodelære*, 4th ed. Tapir, 1985.

[2] R. Walpole, R. Myers, and S. Myers, *Probability and statistics for engineers and scientists*, 6th ed. Prentice-Hall, 1998.

[3] H. Taylor and S. Karlin, *An introduction to stochastic modeling*, 2nd ed. Academic Press, 1994.

[4] V. B. Iversen, *Data- og teletrafikteori.* Den Private Ingeniørfond, Technical University of Denmark, 1999.

[5] G. M. Birtwistle, *DEMOS reference manual.* Bradford, England: Dep. of Computer Science, University of Bradford, 1979.

[6] J. Banks, J. S. Carson, and B. L. Nelson, *Discrete-event system simulation.* Prentice-Hall, 1996.

[7] P. Bratley, B. L. Fox, and L. E. Scharge, *A guide to simulation.* Springer-Verlag, 1987.

[8] P. J. Emstad, *Modellering og måling av teletrafikk.* Department of Computer Science and Telematics, 1996.

[9] B. E. Helvik, *An introduction to the design and evaluation of dependable computing systems and communication networks.* Department of Telematics, NTNU, 1999.

[10] K. Stordahl, "Prognostisering," *Telektronikk*, vol. 90, no. 1, 1994.

[11] ITU-T, *Terms and definitions related to quality of service and network performance including dependability.* ITU-T, August 1994, ITU-T Recommendation E.800.

[12] ISO/IEC JTC 1/SC 6, *Information technology - Quality of service - framework.* ISO/IEC, 1998, international standard ISO/IEC DIS 13236.

[13] ETSI TC-NA, "Network aspects - general aspects of quality of service and network performance," ETSI, Tech. Rep. ETR 003 (Ref:RTR/NA-042102), October 1994.

[14] A. Avizienis, J. C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *Dependable and Secure Computing, IEEE Transactions on*, vol. 1, pp. 11–33, 2004.

[15] ITU-T, *General aspects of quality of service and network performance in digital networks, including ISDN.* ITU-T, 1993, ITU-T Recommendation I.350.

[16] V. B. Iversen, *Handbook in Teletraffic engineering*, ITC / ITU-D SG.2, Ed. ITC / ITU-D, 2005.

[17] L. Kleinrock, *Queueuing systems.* John Wiley & Sons, 1975, vol. 1: Theory.

[18] P. A. W. Lewis and E. J. Orav, *Simulation methodology for statisticians, operations analysts, and engineers.* Wadsworth Publ. Co., 1988, vol. 1.

[19] G. Casella and R. L. Berger, *Statistical inference*, 2nd ed. Duxbury, 2002.

[20] D. Cox, *Renewal theory.* London: Methuen & Co., 1967.

[21] A. S. Tanenbaum, *Computer Networks*, 4th ed. Prentice-Hall, 2003, pp. 700–704.

[22] O. Rose, "Statistical properties of mpeg video traffic and their impact on traffic modeling in ATM systems," University of Würzburg, Institute of Computer Science, Tech. Rep. 101, February 1995.

[23] S. M. Ross, *A course in simulation.* MacMillan, 1990.

[24] G. M. Birtwistle, "DEMOS - a system for discrete event modelling on Simula," 2003. [Online]. Available: http://www.dcs.shef.ac.uk/~graham/research/demos.pdf

[25] G. S. Fishman, *Concepts and methods in discrete event digital simulation.* John Wiley & Sons, 1973.

[26] D. Y. Downham and F. D. K. Roberts, "Multiplicative congruential pseudo-random number generators," *Computer Journal*, vol. 10, no. 1, pp. 74–77, 1967.

[27] D. H. Lehmer, "Mathematical methods in large-scale computing units," in *Proceedings of the Second Symposium on Large Scale Digital Computing Machinery.* Harvard University Press, Cambridge, Mass, 1951, pp. 141–146.

[28] K. M. Olsson, "Some different methods in using markov chains in discrete time applicable to traffic simulation and certain accuracy problems in this context," in *Proceedings of the Fifth International Teletraffic Congress [ITC-5]*, New York, June 14-20 1967.

[29] L. Kosten, "On the measurement of congestion quantities by means of fictious traffic," *Het P.T.T. Bedrijf*, vol. 49, no. 2, 1948.

[30] R. E. Shannon, *Systems simulation - The art and science.* Prentice-Hall, 1975.

[31] B. Feng, "Selection of simulation software for teletraffic research and education," in *11th Nordic teletraffic seminar (NTS-11)*, Stockholm, Sweden, August 1993.

[32] B. Kirkerud, *Object-oriented programming with SIMULA.* Addison-Wesley, 1989.

[33] T. O. Engset, *Die wahrscheinlichkeitsrechnung zur bestimmung der wählerzahl in automatischen fernsprechämtern.* Elektrotechnische Zeitschrift, 1918.

[34] J. R. Jackson, "Networks of waiting lines," *Operations Research*, vol. 5, no. 4, pp. 518–521, Aug 1957.

[35] P. J. Burke, “The output of a queuing system,” *Operations Research*, vol. 4, no. 6, pp. 699–704, Dec. 1956.

[36] S. R. Ali, *Digital switching systems: system reliability and analysis.* McGraw-Hill, 1997.

[37] P. Jalote, *Fault tolerance in distributed systems.* Prentice-Hall, 1994.

[38] B. W. Johnson, *Design & analysis of fault tolerant digital systems.* Addison-Wesley, 1988.

[39] T.-T. Y. Lin and D. P. Siewiorek, “Error log analysis: statistical modeling and heuristic trend analysis,” *IEEE Transactions on Reliability*, vol. 39, no. 4, pp. 419–432, 1990.

[40] J. Gray, “A census of tandem system availability between 1985 and 1990,” *IEEE Transactions on Reliability*, vol. 39, no. 4, pp. 409–418, October 1990.

[41] D. R. Kuhn, “Sources of failure in the public switched telephone network,” *Computer*, vol. 30, no. 4, pp. 31–36, April 1997.

[42] D. Oppenheimer and D. A. Patterson, “Architecture and dependability of large-scale internet services,” *IEEE Internet Computing*, vol. 6, no. 5, pp. 41–49, 2002.

[43] J. A. Buzacott, “Markov approach to finding failure times of repairable systems,” *IEEE Transactions on Reliability*, vol. 19, no. 4, pp. 128–133, November 1970.

[44] R. A. Shaner, K. S. Trivedi, and A. Puliafito, *Performance and reliability analysis of computer systems*, ser. ISBN 0-7923-9650-2. Kluwer Academic Publisher, 1996.

[45] A. Høyland and M. Rausand, *Systems reliability theory: models and statistical methods.* John Wiley & Sons, 1994.

[46] R. E. Barlow and F. Proschan, *Statistical theory of reliability and life testing*, 2nd ed. Silver Springs, Maryland, USA: To begin with, 1981.

Index

Formula sheet

General

X	random variable
Ω	sample space
A_i	partition of Ω

Bayes' formula

$$P(A_i \mid B) = \frac{P(B \mid A_i) \times P(A_i)}{\sum_j P(B \mid A_j) \times P(A_j)} = \frac{P(B \mid A_i) \times P(A_i)}{P(B)} \tag{1}$$

Cumulative distribution function (cdf)

$$F_X(x) = P(X \leqslant x),\ \forall x \in \Omega \tag{2}$$

Probability density function (pdf) (X continuous)

$$f_X(x) = \lim_{\mathrm{d}x \to 0} \frac{P(x < X \leqslant x + \mathrm{d}x)}{\mathrm{d}x},\ \forall x \in \Omega \tag{3}$$

Probability mass function (pmf) (X discrete)

$$f_x = P(X = x),\ \forall x \in \Omega \tag{4}$$

j^{th}order **moment** (X continuous)

$$m_j = \mathrm{E}(X^j) = \int_\Omega x^j f(x)\,\mathrm{d}x,\ j \in \mathbb{N}^* \tag{5}$$

j^{th}order **central moment** (X continuous)

$$\mathrm{E}((X-\mu)^j) = \int_\Omega (x-\mu)^j f(x)\,\mathrm{d}x,\ j \in \mathbb{N}^* \tag{6}$$

Palm's identity (X continuous, non-negative and real-valued)

$$m_j = \mathrm{E}(X^j) = \int_0^\infty x^j f(x)\,\mathrm{d}x = \int_0^\infty j x^{j-1}(1 - F(x))\,\mathrm{d}x,\ j \in \mathbb{N}^* \tag{7}$$

Expected value of X

$$\mu = \mathrm{E}(X) \tag{8}$$

Variance of X

$$\sigma^2 = \mathrm{Var}(X) = E_X\left((X-\mu)^2\right) \tag{9}$$

Standard deviation

$$\sigma = \sqrt{\mathrm{Var}(X)} \tag{10}$$

Coefficient of variation

$$\rho = \frac{\sigma}{\mu} = \frac{\sqrt{\mathrm{E}(X^2) - \mathrm{E}(X)^2}}{\mathrm{E}(X)} \tag{11}$$

Expected value of a sum of random variables

$$\mathrm{E}\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n} \mathrm{E}(X_i) \tag{12}$$

Variance of a sum of mutually independent random variables

$$\mathrm{Var}\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n} \mathrm{Var}(X_i) \tag{13}$$

Variance of a sum of non mutually independent random variables

$$\mathrm{Var}\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n}\sum_{j=1}^{n} \mathrm{Cov}(X_i, X_j) \tag{14}$$

Covariance

$$\sigma_{X,Y} = \mathrm{Cov}(X,Y) = \mathrm{E}((X-\mu_X)(Y-\mu_Y)) = \mathrm{E}(XY) - \mu_X\mu_Y \tag{15}$$

Table : Characteristics of relevant distributions

$X \sim$	$f(x)$	$F(x)$	$\mathrm{E}(X)$	$\mathrm{Var}(X)$
n.e.d.	$\lambda e^{-\lambda x}$	$1 - e^{-\lambda x}$	$\frac{1}{\lambda}$	$\frac{1}{\lambda^2}$
Erlang-k	$\frac{(\lambda x)^{k-1}}{(k-1)!}\lambda e^{-\lambda x}$	$1 - \sum_{j=0}^{k-1} \frac{(\lambda x)^j}{j!} e^{-\lambda x}$	$\frac{k}{\lambda}$	$\frac{k}{\lambda^2}$
Gamma	$\frac{(\lambda x)^{k-1}}{\Gamma(k)}\lambda e^{-\lambda x}$	$\frac{1}{\Gamma(k)} \int_0^{\lambda x} u^{k-1} e^{-u}\,\mathrm{d}u$	$\frac{k}{\lambda}$	$\frac{k}{\lambda^2}$
Weibull	$k\lambda(\lambda x)^{k-1} e^{-(\lambda x)^k}$	$1 - e^{-(\lambda x)^k}$	$\frac{1}{\lambda}\Gamma\left(\frac{1}{k}+1\right)$	$\frac{1}{\lambda^2}\left(\Gamma\left(\frac{2}{k}+1\right) - \Gamma^2\left(\frac{1}{k}+1\right)\right)$
Uniform	$\frac{1}{b-a}$	$\frac{x-a}{b-a}$	$\frac{a+b}{2}$	$\frac{(b-a)^2}{12}$
Standard Normal, $\mathcal{N}(0,1)$	$\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$	$\Phi(x) = \int_{-\infty}^{x} \phi(u)\,\mathrm{d}u$	0	1
Normal, $\mathcal{N}(\mu,\sigma^2)$	$\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$	$\Phi\left(\frac{x-\mu}{\sigma}\right)$	μ	σ^2
Bernoulli	$\begin{cases} p & x=1 \\ 1-p & x=0 \end{cases}$		p	$p(1-p)$
Geometric	$(1-p)^x p$	$1-(1-p)^{x+1}$	$\frac{1-p}{p}$	$\frac{1-p}{p^2}$
Shifted geometric	$(1-p)^{x-1} p$	$1-(1-p)^x$	$\frac{1}{p}$	$\frac{1-p}{p^2}$
Binomial	$\binom{n}{x} p^x (1-p)^{n-x}$	$\sum_{i=0}^{x} \binom{n}{i} p^i (1-p)^{n-i}$	np	$np(1-p)$
Poisson	$\frac{\alpha^x}{x!} e^{-\alpha}$	$e^{-\alpha} \sum_{i=0}^{x} \frac{\alpha^i}{i!}$	α	α

Covariance (X and Y continous)

$$\sigma_{X,Y} = \int_\Omega \int_\Omega ((x-\mu_X)\cdot(y-\mu_Y))f(x,y)\,\mathrm{d}x\,\mathrm{d}y \tag{16}$$

Covariance (X and Y discrete)

$$\sigma_{X,Y} = \sum_x \sum_y ((x-\mu_X)\cdot(y-\mu_Y))f_{x,y} \tag{17}$$

Correlation coefficient

$$\rho_{X,Y} = \frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{Var}(X)\cdot\mathrm{Var}(Y)}} = \frac{\sigma_{X,Y}}{\sigma_X\sigma_Y},\ -1 \leqslant \rho_{X,Y} \leqslant 1 \tag{18}$$

Intensity of a point process $X(t)$

$$z(t) = \lim_{\Delta t\to 0}\frac{\mathrm{E}(N(t+\Delta t)-N(t))}{\Delta t} \tag{19}$$

Rate of a point process $X(t)$

$$\phi(y) = \lim_{\Delta y\to 0}\frac{P(y < Y \leq y+\Delta y \mid Y > y)}{\Delta y} \tag{20}$$

Autocovariance of a stationary process $X(t)$

$$\sigma_{XX}(\tau) \equiv \sigma_{X(t)X(t+\tau)} = \mathrm{E}(X(t)X(t+\tau)) - \mu^2 \tag{21}$$

Autocorrelation

$$\rho_X(\tau) = \frac{\sigma_{XX}(\tau)}{\sigma_{XX}(0)} \tag{22}$$

Residual lifetime

$$f(t+a \mid a) = \frac{f(t+a)}{1-F(a)} \tag{23}$$

Forward recurrence time

$$f_T(t) = \frac{P(t < T \leqslant t+\mathrm{d}t)}{\mathrm{d}t} = \int_t^\infty \frac{1}{x}\frac{1}{\mathrm{E}(X)}x f_X(x)\,\mathrm{d}x = \frac{1-F_X(t)}{\mathrm{E}(X)} \tag{24}$$

Expected forward recurrence time

$$\mathrm{E}(T) = \frac{\mathrm{E}(X^2)}{2\mathrm{E}(X)} \tag{25}$$

Regular point process $N(t)$ is the number of events and S_r the time to the r^{th} event.

$$P(N(t) < r) = P(S_r > t) \tag{26}$$

Renewal function

$$H(t) = \mathrm{E}(N(t)) = \frac{t}{\mathrm{E}(Y)} + \frac{\mathrm{Var}(Y) - \mathrm{E}(Y)^2}{2\mathrm{E}(Y)^2} + o(1) \tag{27}$$

Traffic

Little's formula

$$\bar{N} = \bar{\lambda} \cdot \hat{W} \tag{28}$$

Offered traffic

$$A = \sum_{i=0}^{\infty} \lambda_i \, p_i \big|_{n=\infty} \mathrm{E}(X) \tag{29}$$

Carried traffic

$$A' = \sum_{i=0}^{k} \min(i, n) p_i \tag{30}$$

Carried traffic $I(t)$ is the observed resource utilisation over the time interval (t_1, t_2)

$$A' = \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} I(t) \, \mathrm{d}t \tag{31}$$

Erlang's B-formula

$$E_n(A) = \frac{A^n/n!}{\sum_{\nu=0}^{n} A^\nu/\nu!} \tag{32}$$

Recursive Erlang's B-formula

$$E_n(A) = \frac{AE_{n-1}(A)}{n + AE_{n-1}(A)}, \; n = 1, 2, \cdots, n \tag{33}$$

Time congestion in an Engset's loss system

$$E_{n,s}(\beta) = p_n = \frac{\binom{s}{n}\beta^n}{\sum_{\nu=0}^{n}\binom{s}{\nu}\beta^\nu},\ s \geqslant n,\ \beta = \frac{\lambda}{\mu} \tag{34}$$

Call congestion in an Engset's loss system

$$B_{n,s}(\beta) = \frac{(s-n)\binom{s}{n}\beta^n}{\sum_{\nu=0}^{n}(s-\nu)\binom{s}{\nu}\beta^\nu},\ s \geqslant n,\ \beta = \frac{\lambda}{\mu} \tag{35}$$

Erlang's C-formula (probability of waiting in an $M/M/n$ system)

$$E_{2,n}(A) = P(W > 0) = p_n \frac{n}{n-A} = \frac{\frac{A^n}{n!}\frac{n}{n-A}}{\sum_{i=0}^{n-1}\frac{A^i}{i!} + \frac{A^n}{n!}\frac{n}{n-A}},\ A < n \tag{36}$$

Expected sojourn time in an $M/M/n$ system

$$\mathrm{E}(S) = \frac{1}{(n-A)\mu}\left(E_{2,n}(A) + n - A\right),\ A < n \tag{37}$$

Waiting time distribution for all customers in an $M/M/n/\infty/FIFO$ system

$$F_W(t) = 1 - E_{2,n}(A)e^{-(n-A)\mu t},\ t \geqslant 0,\ A < n \tag{38}$$

Jackson network

$$\Lambda_k = \lambda_k + \sum_{j=1}^{K} p_{jk}\Lambda_j,\ k,j = 1,\ldots,K \tag{39}$$

Dependability

Mean Time to First Failure

$$MTFF = \mathrm{E}(T_{FF}) = \int_0^\infty R(t)\,\mathrm{d}t \tag{40}$$

Mean Time to Catastrophic Failure

$$MTCF = \mathrm{E}(T_{CF}) \tag{41}$$

Mean Time Between Failures

$$MTBF = \mathrm{E}(T_{BF}) \tag{42}$$

Mean Up Time

$$MUT = \mathrm{E}(T_U) \tag{43}$$

Mean Down Time

$$MDT = \mathrm{E}(T_D) \tag{44}$$

Mean Time To Failure

$$MTTF = \mathrm{E}(T_F) = \int_0^\infty \tilde{R}(t)\,\mathrm{d}t \tag{45}$$

Mean Time Between Failures calculated from a state transition diagram

Ω_W	set of all working states
Ω_F	set of all failed states
q_{ij}	transition intensity from state i to state j
p_i	stationary probability that the system is in state i

$$\frac{1}{MTBF} = \sum_{i\in\Omega_W}\sum_{j\in\Omega_F} p_i \cdot q_{ij} = \sum_{i\in\Omega_W}\sum_{j\in\Omega_F} p_j \cdot q_{ji} \tag{46}$$

Mean Time to First Failure for a system that cannot be repaired and a structure of indenpendent elements with constant failure rate λ_i ($\lambda = \lambda_i, \forall i$).

$$MTFF_{\text{series}} = \left(\sum_{i=1}^{n} \lambda_i\right)^{-1} \tag{47}$$

$$MTFF_{\text{parallel}} = \frac{1}{\lambda}\sum_{i=1}^{n}\frac{1}{i} \tag{48}$$

$$MTFF_{k\text{−out−of−}n} = \frac{1}{\lambda}\sum_{j=k}^{n}\frac{1}{j} \tag{49}$$

Reliability function

$$R(t) = P(T_{FF} > t), \text{ service delivery starts when the system is new} \tag{50}$$

$$\tilde{R}(t) = P(T_F > t), \text{ service delivery starts while the system is stationary} \tag{51}$$

$$R_{\text{series}}(t) = \prod_{i=1}^{n} R_i(t) \tag{52}$$

$$R_{\text{parallel}}(t) = 1 - \prod_{i=1}^{n}(1 - R_i(t)) \tag{53}$$

$$R_{k-\text{out}-\text{of}-n}(t) = \sum_{i=k}^{n} \binom{n}{i} R(t)^i (1 - R(t))^{n-i}, \; R_i(t) = R(t), \; \forall i \tag{54}$$

Let $I(t) = \begin{cases} 1 & \text{the system is up at time } t \\ 0 & \text{otherwise} \end{cases}$

Instantaneous availability

$$A(t) = P(I(t) = 1) \tag{55}$$

Interval availability between t_1 and t_2

$$A(t_1, t_2) = \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} A(t)\,\mathrm{d}t = \mathrm{E}\left(\frac{1}{t_2 - t_1} \int_{t_1}^{t_2} I(t)\,\mathrm{d}t\right) \tag{56}$$

(Asymptotic) Availability

$$A = \lim_{t\to\infty} P(I(t) = 1) = \lim_{t\to\infty} \mathrm{E}(I(t)) = \frac{MUT}{MDT + MUT} = \frac{MUT}{MTBF} \tag{57}$$

$$A = \sum_{i\in\Omega_W} p_i \tag{58}$$

where Ω_W is the set of all working states and p_i the stationary probability for state i.

$$A_{\text{series}} = \prod_{i=1}^{n} A_i \tag{59}$$

$$A_{\text{parallel}} = 1 - \prod_{i=1}^{n}(1 - A_i) \tag{60}$$

$$A_{k-\text{out}-\text{of}-n} = \sum_{i=k}^{n} \binom{n}{i} A^i (1 - A)^{n-i}, \; A_i = A, \; \forall i \tag{61}$$

Unavailability

$$U(\cdots) = 1 - A(\cdots) \tag{62}$$

Estimators

Let Θ be an *unbiased* and *consistent estimator* of a *parameter* ξ, i.e. $\mathrm{E}(\Theta) = \xi$ and $\mathrm{Var}(\Theta(n)) \to 0$ when $n \to \infty$. Let $X_1, X_2, \cdots, X_n$ be n independent and identically distributed observations, and $\mathrm{E}(X_i) = \xi$ and $\mathrm{Var}(X_i) = \sigma^2,\ i = 1, 2, \cdots, n$.

Time average

$$\bar{X} = \frac{1}{T}\int_0^T X(t)\,\mathrm{d}t \tag{63}$$

Sample mean

$$\hat{X} = \frac{1}{n}\sum_{i=1}^{n} X_i \tag{64}$$

Sample variance

$$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \hat{X})^2 = \frac{1}{n-1}\sum_{i=1}^{n} X_i^2 - \frac{n}{n-1}\hat{X}^2 \tag{65}$$

Variance of the sample mean

$$S_{\hat{X}}^2 = \frac{S^2}{n} \tag{66}$$

Standard error of the sample mean

$$S_{\hat{X}} = \frac{S}{\sqrt{n}} \tag{67}$$

$1 - \alpha$ **confidence interval** for $\hat{X}$ (with unknown variance)

$$\left(\hat{X} - t_{\alpha/2,n-1}\frac{S}{\sqrt{n}}, \hat{X} + t_{\alpha/2,n-1}\frac{S}{\sqrt{n}}\right) \tag{68}$$

($t_{\alpha/2,n-1}$ is the $\alpha/2$-quantile of the Student's t-distribution with $n-1$ degrees of freedom.)

Some mathematical formulas

Exponential function

$$e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!} \tag{69}$$

$$e^x = \lim_{n\to\infty} \left(1 + \frac{x}{n}\right)^n \tag{70}$$

Binomial theorem

$$\sum_{k=0}^{n} \binom{n}{k} a^{n-k} b^k = (a+b)^n,\ n > 0 \tag{71}$$

Geometric series

$$\sum_{k=0}^{n} x^k = \frac{1 - x^{n+1}}{1 - x},\ n \in \mathbb{N},\ |x| < 1 \tag{72}$$

De Morgan's theorem

$$\overline{A + B} = \overline{A} \cdot \overline{B} \tag{73}$$

$$\overline{A} + \overline{B} = \overline{A \cdot B} \tag{74}$$