Chapter 7
Process Analysis

Abstract In this chapter, we present analysis techniques for business processes. This includes process analysis and simulation, process performance management, and process mining.

7.1 Introduction and Terminology

Process analysis techniques are used to answer process-related questions and to assess key performance indicators such as throughput times for process instances. In particular, process analysis techniques can be used to evaluate *qualitative* as well as *quantitative* analysis questions over process models and process instances.

Qualitative questions, in general, address the structure, behavior, and quality of business process models on the one side and qualitative execution aspects of process instances on the other side. According to [6], qualitative questions refer to the identification of unnecessary parts or value-added steps in the process. In addition, qualitative process analysis subsumes process *verification* [31], i.e., formulating and checking the soundness of process models and process instances. Note that the properties that are to be verified partly depend on the business process modeling language (cf. Chap. 2).

Basically, we distinguish between structural, behavioral, and semantic soundness in business processes [17, 38]. Which criteria must be fulfilled in order to guarantee soundness partly depends on the underlying process meta-model [23]. The structural soundness of a Petri net, for example, requires a bipartite structuring of the process graph. Moreover, certain properties of the process graph structure, e.g., connectedness or uniqueness of process start and end nodes, might be demanded.

Behavioral soundness implies—among other properties—that no transitions are dead, i.e., they cannot fire anymore, except for the final state where the end state is marked with a token [38].

In this book, we assume process models that are structurally and behaviorally sound. How to check semantic soundness will be discussed in this chapter in the context of business process compliance.

Semantic soundness can be approached from two angles: first of all, semantic correctness might imply the question of how well a process model reflects reality. This question can be answered by process *validation*. Validation techniques for

W. Grossmann, S. Rinderle-Ma, *Fundamentals of Business Intelligence*,
Data-Centric Systems and Applications, DOI 10.1007/978-3-662-46531-8_7

comparing process models and real-world process executions by means of *conformance checking* will be presented in Sect. 7.4.3.

Secondly, semantic soundness includes the verification of semantic properties over business processes that are usually captured by semantic process constraints. It is referred to by the term *business process compliance* [17].

BPM tools often include qualitative process analysis functionalities to check, for example, for business process flaws, such as media breaks[1] [3], or activities without any assigned actor within a process model. Commercial BPM tools, e.g., ARIS Platform,[2] offer various ways of documenting the qualitative properties of process models such as organizational handbooks. One aim of qualitative process analysis is to redesign the process model in a leaner or more efficient way. Redesign measures can include the reordering of process activities (e.g., parallelization) or centralization of resources. A collection of best practices in business process redesign can be found in [19].

Quantitative process analysis is complementary to qualitative process analysis as "results obtained from qualitative analysis are sometimes not detailed enough to provide a solid basis for decision making" [6]. If qualitative analysis yields, for example, that a certain part of the process model probably constitutes a bottleneck in the process execution, this result can be underpinned by quantitatively analyzing the process models by determining process performance measurements.

Figure 7.1 shows the life cycles of business processes (BP) and automated processes (AP), i.e., workflows. A business process design and automation project is often divided into two subprojects, i.e., firstly a project for acquiring, designing, and optimizing the business processes of interest and secondly a process automation project that designs the automated processes, implements and configures them, creates and starts the executed process instances, and diagnoses these running processes.

It is important to understand that the analysis techniques for BP operate on BP models, whereas analysis techniques for AP operate on real-world execution data of processes, e.g., on event logs. Note that for all phases of the BP life cycle, quantitative as well as qualitative questions might arise. Examples for a qualitative and quantitative question are summarized in Table 7.1.

The following sections present analysis techniques for the different phases of BP and AP life cycles. Section 7.2 introduces techniques for BP analysis and simulation as well as optimization. Section 7.3 illustrates techniques that are used for measuring the performance of running process instances and for analyzing process execution data within a process warehouse. Section 7.4 introduces process mining as a collection of techniques to analyze processes during runtime (*online*)

[1] Media breaks occur, for example, if computerized data is printed, transferred as paper-based form via some process steps, and later inserted into a computerized format again.

[2] http://www.softwareag.com/corporate/products/aris_platform/default.asp.

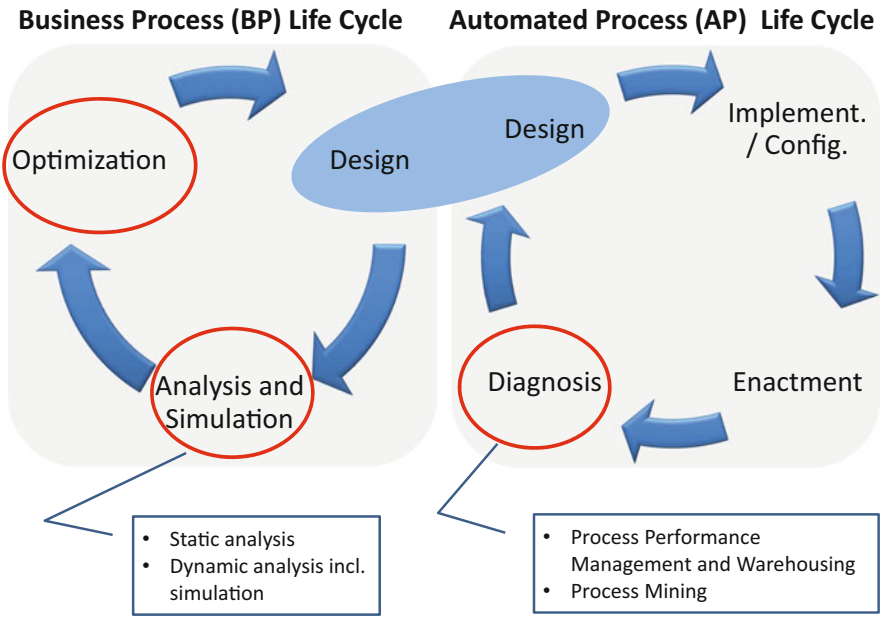


Fig. 7.1 Analysis phases within business process (BP) and automated process (AP) life cycle, cf. [9]

Table 7.1 Examples for qualitative and quantitative questions during different phases of BP and AP life cycle

BP Analysis and Simulation	*Which activities are not assigned a role? (qualitative)*
	How long is the average duration of an activity XY based on simulation? (quantitative)
AP Diagnosis	*Given a set of process instance executions, how does the underlying process model look like? (qualitative)?*
	How many running process instances currently exceed the deadline? (quantitative)?

or *ex post* (*offline*). While Sects. 7.2–7.4 refer to the analysis of process models and process instances, Sect. 7.5 draws on business process compliance and includes business rules and constraints as additional aspects.

7.2 Business Process Analysis and Simulation

Before automating business processes within *process-aware information systems* (PAIS), they are often modeled, analyzed, and optimized at a semantically higher level using specification languages such as BPMN or EPCs (cf. Chap. 2). The typical

goals of business process analysis are the reduction of errors and the optimization of key performance indicators such as process throughput time. Two example questions have been formulated in Table 7.1:

Which activities are not assigned a role? (qualitative)
How long is the average duration of activity XY based on simulation? (quantitative)

A first distinction for choosing an analysis technique is what input is used for the analysis. We can distinguish between analysis that is based on the business process model only (*static analysis*) and analysis that uses the process model and simulation data (*dynamic analysis*).

7.2.1 Static Analysis

Static analysis refers to all techniques that are applied at process model level, i.e., without taking into consideration information on process executions in terms of simulation data or real process instance execution data. An example is process cost calculation as depicted in Fig. 7.2. The estimated costs for each process activity together with the probability assigned at the outgoing paths of the XOR branching can be aggregated to the expected overall process costs.

Tools such as ARIS Business Process Analysis Platform[3] enable the aggregation of numeric values (mainly time and costs) at process model level.

7.2.2 Dynamic Analysis and Simulation

Dynamic analysis aims at predicting the behavior of the process during execution and is based on *simulation* data. Simulation refers to the artificial creation and execution of process instances taking into consideration assigned parameters. The simulation of processes does not involve their execution, i.e., no actual data is written or read, no work lists are created, and no activity programs are invoked.

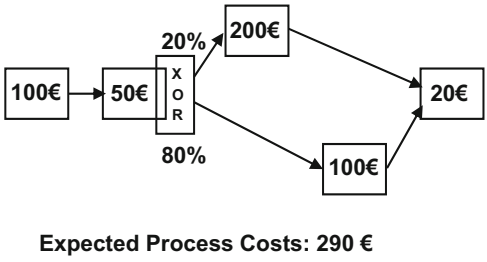


Fig. 7.2 Process cost calculation (example)

[3] http://www.softwareag.com/corporate/products/aris/bpa/overview/default.asp.

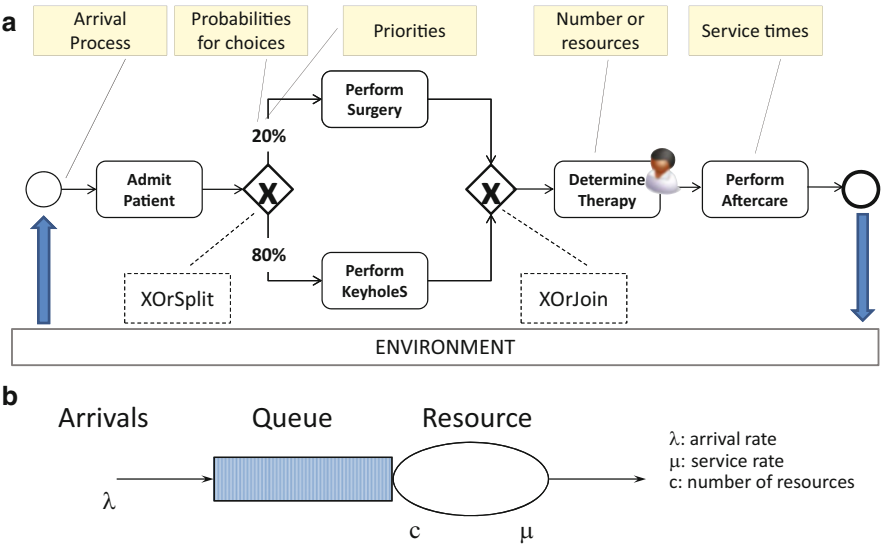


Fig. 7.3 (**a**) Simulation setup and (**b**) modeling instance arrival based on waiting queues; figure based on [32]

The main goal of a process simulation is the quantitative evaluation of the process model. Typical input parameters for process simulation include:

Input Parameters for Business Process Simulation:

- The number of simulated processes
- Probabilities at alternative branchings
- Capacities of users and resources
- Strategies to resolve bottlenecks
- Costs, processing, and transport durations (some tools enable the definition of discrete values but also the definition based on probability distributions)

Figure 7.3 illustrates a typical process simulation setup [32] including the parameters as mentioned above. It is also of interest to consider the arrival process of process executions coming from the environment, produced, for instance, by some other process. In the treatment example, this arrival process could reflect patients being referred from the general practitioner to a specialist. The arrival process of the process instances can be simulated based on *waiting queues* [32], not only at the start point of a process but also at every other process activity [21].

Several BPM tools offer process simulation functionality, for example, Bonapart, ARIS Business Process Analysis Platform, and Signavio.

Typically, the simulation function realized in BPM tools enables the definition of several simulation parameters for process activities. Consider the treatment example displayed in Fig. 7.3, specifically, the process activity `determine therapy`.

Typically, based on the BPM tool, users would be assigned to this activity at first. For users, it is mostly possible to deposit unavailability times and processing strategies that are taken into consideration during the simulation. Connecting the process perspective with the organizational perspective allows to analyze the resource utilization and therefore might be helpful in detecting bottlenecks. Further on, in most cases, it is possible to define processing times for activities based on probability distributions such as normal distribution with a certain mean and standard deviation. This simulates that for many process activities, realistically, no fixed processing/execution time can be determined, but rather experience data with a deviation. At branching points, probabilities for selecting the different outgoing paths can be determined in a discrete way or based on probability distributions. Furthermore, usually at the starting points of the process, it can be set how many incoming processes will be generated. For example, this may be based on a fixed value within a given time frame or on a probability distribution.

During simulation, processing times are determined on the grounds of this probability distribution. Note that simulation is mostly accompanied by an *animation*, i.e., the visual reflection of the process execution ("blinking process activities"). A process simulation typically yields simulation protocols as well as different aggregations, i.e.:

Expected Output from Business Process Simulation:

- Process duration under given assumptions (e.g., a defined number of process executions)
- The number of possible process executions in a given time frame
- The workload of resources
- The localization of bottlenecks and flaws within the process model

Some BPM tools offer graphical support in order to visualize the simulation results. However, the interpretation of what the simulation results actually mean, e.g., where will the process bottlenecks occur during runtime, remains still a task for the (human) process analyst.

In addition to in-built simulation functionalities as offered by BPM tools, simulation scenarios can be set up by using process simulation environments such as colored Petri net (CPN) Tools[4] [32]. CPN tools provide a powerful environment for modeling and simulating processes. Due to a certain complexity in specifying the models and parameters, other simulation environments for specific analysis purposes have been developed recently. Examples include SecSy [1] for the simulation of processes with respect to security-relevant aspects or $DPA_{Sim}^{TimeSeries}$ [11] for the simulation of times series data in process applications.

[4] http://cpntools.org/.

7.2.3 Optimization

The previous sections provided an overview on *how* business processes can be analyzed. The *what* has been addressed in an exemplary way. Literature mostly names four dimensions of a business process which can be analyzed and optimized [19, 24]: cost, time, quality, and flexibility.

Optimizing all four dimensions at the same time is hard or even impossible. Reducing, for example, the throughput time of the process execution might result in higher costs or a lower quality of the products [20].

Cost and time are quantitative dimensions. Hence, they can be analyzed using static and dynamic analysis including simulation. It is much harder to analyze quality and flexibility as they are qualitative dimensions. One possibility would be to find appropriate quantifications for quality and flexibility. Appropriate means that the quantification must yield meaningful results when applying static or dynamic analysis.

How can a business process be optimized? In [20], several best practices for business process optimizations are summarized that have been also evaluated with practitioners. Structural optimizations might include eliminating, combining, and reordering activities.

An activity might be eliminated, for example, if it has been executed only for historical reasons, e.g., archiving a certain paper form that is available in an online archive anyway. Combining activities is reasonable if their combined execution can be handled more efficiently than in a separated manner. Reordering activities refers to sequentializing parallel activities or parallelizing sequentially ordered activities. Reordering can also mean to reorder the process.

Moreover, there are optimizations that regard the organizational perspective of a business process such as empowering actors, going from specialists to generalists or vice versa, and the reduction or extension of organizational resources.

Empowering actors can mean to integrate them better into decisions. The reduction of organizational resources probably leads to a reduction of costs, their extension to an increase of costs. In turn, the extension of resources might result in decreased throughput times or increased quality and flexibility.

Finally, there are best practices that concern the overall process such as its integration with another process.

Overall, there is no automatism for optimizing business processes, i.e., there are no rules that say *if you detect this kind of flaw in the business process, you apply that kind of optimization*. Mostly, business process optimization or redesign is a "trial-and-error" procedure, i.e., applying a best practice and try to evaluate the success.

For some of the best practices, the success of applying them can be measured more easily than for others. For some of them, we can even analyze their success at business process model level, i.e., before implementing and executing the process in real life. Examples include structural optimizations and the reduction/extension

of organizational resources. For these best practices, static and dynamic analysis including simulation can be used for the original business process (AS-IS model) and the optimized one (TO-BE model). The results can be compared, and the success of the evaluation can be evaluated. Some BPM tools offer support for this variant management.

7.2.4 Summary: Process Analysis and Simulation

In many practical BI applications, process models are developed based on a small set of available data, for example, interviews, workshops, or guidelines. In order to understand how these process models will "behave" during runtime, often, process simulation is used. In this section, we described how to set up simulation models and how to interpret the resulting data. Further, we try to raise awareness that process simulation and process execution are two different things. Simulation creates artificial data, whereas process execution creates real-world data. How to analyze the latter will be discussed in the following sections.

7.3 Process Performance Management and Warehousing

Quantitative process analysis techniques such as simulation and queuing are applied during the process design phase (cf. Fig. 7.1), i.e., they include process model information and create artificial process execution data. Hence, as stated in [31], a simulation mimics reality, but not vice versa.

7.3.1 Performance Management

Process performance management provides means to analyze real-world process execution data in a quantitative way. More precisely, typical process performance analysis tasks are process monitoring, monitoring of key performance indicators defined on process executions (e.g., throughput time), and analysis of further aspects such as organizational structures.

A challenge in this context is the provision of process execution data in real time (cf. Chap. 3) in such a way that the above tasks can be executed in a satisfactory way. For process monitoring, most process execution engines (also referred to as workflow engines) provide monitoring components that visualize the current execution states of process instances, e.g., the *process monitor* of the AristaFlow

BPM Suite[5] or the monitoring component of the Cloud Process Execution Engine (CPEE).[6]

Some of the tools extend the monitoring function with the specification and monitoring of key performance indicators (e.g., throughput time) and functionalities. The latter include sending alerts if a certain KPI is exceeded or a defined deadline is going to be violated. This extended process monitoring component is often referred to as process cockpit or process dashboard. Tools for process performance management include ARIS Performance Manager (ARIS PPM) and IBM Business Monitor.

In IBM Business Modeler Advanced Version 7.0, for example, KPIs can be either chosen from a catalog of predefined KPIs or defined individually. For visualizations of KPIs, we refer to Chap. 4. The provided KPI catalog is categorized into different domains such as supply chain planning with associated KPIs. In addition, it is possible to define instance measures that can be monitored through a dashboard as well as by alerts.The latter inform users about certain situations during runtime, e.g., a task duration exceeds a certain threshold. Aggregated measures can be calculated based on instance measures. An example is the aggregated duration for a certain task over all instance executions.

The provision of process execution data in realtime remains a major challenge for the application of these tools, particularly, if the processes span multiple systems and application components. Hence, some tools demand for some kind of closed-world assumption, i.e., that the processes are executed within one environment, preferably the one provided by the tool itself, following a specific modeling language. Alternatively, the user is burdened with the task of linking process information from different sources and hence establishing an integrated process data basis.

7.3.2 Process Warehousing

As for data warehousing, process warehousing aims at the offline analysis of process execution data based on a table format. For an example, consider the data structure for process warehouse analysis as illustrated by Fig. 3.13 (cf. Sect. 3.4.1).

This multidimensional structure can be implemented in different ways by using, for instance, a star or snowflake schema as described in Sect. 3.4.1. Based on multidimensional structures, different analyses such as OLAP and data mining can be carried out:

- *Online analytical processing (OLAP)* techniques are described in detail in Sect. 3.4.1. Conceptually, there is no difference in applying OLAP techniques to data warehouse or process warehouse data.

[5] www.aristaflow.com.

[6] cpee.org.

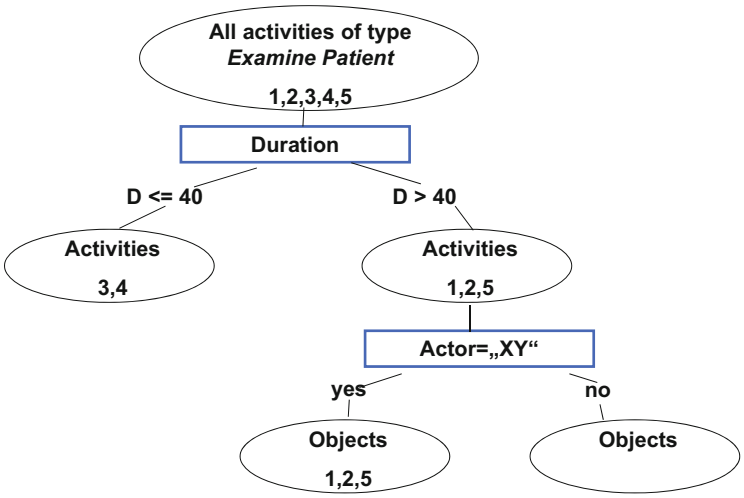


Fig. 7.4 Mining process warehouse data: classification by decision trees [11]

- *Data mining* techniques foster the extraction of implicit and not-yet-known hypotheses from data. Several data mining techniques have been explained in Chaps. 5 and 6. In the following, we want to show how these techniques can be specifically applied in the context of multidimensional process data.

Not much investigation exists on how to apply data mining techniques to process warehouse data. One example is the application of classification, more precisely, decision trees, that can be used for the analysis of exceptions in process executions such as overlong duration of activities [11].

`EBMC`^{2} `Use Case: Process Warehouse Analysis Applying Decision Trees`

Consider the process warehouse setting of the EBMC^2 project as depicted in Fig. 3.9. Figure 7.4 shows a (artificial) decision tree that could have been generated based on this data. Analyzing the duration of the task `Examine Patient`, a distinction could be drawn between process activities of duration below and above a value of 40. For the activities of longer duration, i.e., activities 1, 2, and 5, it could be concluded that all of these activities have been executed by actor `XY`. Note that such results can be critical if the performance of certain persons is evaluated. Hence, they may even be contradictive with the works council. In this case, the analysis can rather be conducted on groups of person or with anonymized data. Details on the EBMC^2 project can be found on the homepage of the book:
www.businessintelligence-fundamentals.com

Decision trees constitute one of the few techniques that have been proposed in connection with process warehouse data. Some approaches suggest the combined application of process mining and data mining [11, 31]. As an example, we will

explain the application of decision trees in connection with process mining analysis in Chap. 8.

7.3.3 *Summary: Process Performance Management and Warehousing*

Overall, the real-time monitoring and analysis of process execution data is of great importance in order to control running process instances. The main focus has been put on the specification and supervision of KPIs with associated warnings and alerts. Less attention has been turned to the analysis of real-time data. Apart from the definition of KPIs and specific techniques to analyze real-time data, the visualization of the running process instances and the KPIs plays an important role. Usually, the user monitors running process instances via a so-called process cockpit or dashboard where, for example, traffic light or steering wheel metaphors are used to convey the information to the user. As visualizations might reveal certain disadvantages for monitoring purposes, such as screen size or the complexity of the data [14], a combination with additional methods, e.g., sonifications, is currently investigated [13]. This might become particularly interesting in environments with huge process execution data (often stored and processed based on events). One example for such an environment is the manufacturing domain which has been investigated within the EU FP7 project ADVENTURE.[7]

Creating and maintaining process warehouse data can be an interesting approach for analyzing process data in an offline fashion. It can be of particular interest to integrate the process warehouse data with other warehouse data in the business context as proposed in [5]. A possible drawback of the process warehouse approach is the possibly "unnatural" multidimensional structuring of process data. Processes are living structures that are executed. Hence, a more execution-oriented modeling and analysis of the data in the form of logs is probably more helpful.

In summary, we can distinguish between the analysis of process execution data in an online or offline mode [33]. Further, we can distinguish between analyzing process data in table or log format.

7.4 Process Mining

In essence, process mining basically addresses two questions: *process discovery* and *conformance checking* [60]. Note that log repair is mentioned as the third question in this reference. However, we will abstain from this aspect in this book.

[7] http://www.fp7-adventure.eu/.

Given the execution data of a set of process instances, the goal of process discovery is to derive the underlying process model, i.e., the process model that was used to create, initiate, and execute those instances reflected by the log. Doing so can be of high interest for many practical applications where processes are executed. This does not always happen explicitly through a process management or workflow system, but across multiple information systems such as database management systems, document management systems, or ERP systems. From an algorithmic perspective, process discovery techniques take process execution data as input and produce a process model as output. The process model can be expressed by any kind of process description language such as Petri nets (cf. Chap. 2). Process discovery can be estimated as unsupervised learning in case there is no reference process model to compare the results of the discovery.

Conformance checking assumes the existence of some process model and checks whether associated log data follows or deviates from this process model. Hence, conformance checking can be subsumed as supervised learning. Typical analysis questions for conformance checking are "where do actors deviate from the process model" or "to what degree does reality reflect the prescribed process structure?"

7.4.1 Process Discovery

Process discovery is an analysis technique based on process execution logs, i.e., it analyzes process instances that have already been executed. Let us start with a short recap from Sect. 3.4.2: Process execution logs store information related to process execution, mostly in terms of events. As a minimum requirement to conduct process discovery techniques, events on process task execution (e.g., start or end events) must be contained within the log together with ordering information which can be either provided by time stamps or by the actual ordering in the log. In order to distinguish events of different process runs or instances, either one log does only contain information of one instance execution or the log entries (events) must be additionally equipped with (unique) identifiers for each instance. Events may be further augmented with information on, for example, the actors who performed a certain task. How this additional information can be exploited and analyzed is discussed in Sect. 8.2.

As process mining has significantly matured during the last years, a variety of process discovery techniques exists nowadays. In this section, we start with explaining the principles of the α-algorithm [4] as it builds the basis for understanding discovery techniques that ground on frequency counting over execution logs. The result of applying the α-algorithm to an event log is a Petri net that represents the underlying process model.

The *heuristic miner* [36] will be sketched as an advancement of the α-algorithm. Finally, we discuss the basic idea of the *genetic miner* [5] that constitutes another

a

Case1: <A, C, D, E, F, G>

Case2: <A, B, D, E, F, G>

Case3: <A, C, D, F, E, G>

Case4: <A, B, D, F, E, G>

Case5: <A, B, D, E, F, G>

b

- *Sequence*: $a \rightarrow_W b \Leftrightarrow a >_W b \wedge \neg (b >_W a)$

- *Exclusive*: $a\#_W b \Leftrightarrow \neg (a >_W b \vee b >_W a)$

- *Parallel*: $a||_W b \Leftrightarrow a >_W b \wedge b >_W a$

c

Sequential: $A \rightarrow_W B$, $A \rightarrow_W C$, $B \rightarrow_W D$, $C \rightarrow_W D$, $D \rightarrow_W E$, $D \rightarrow_W F$, $E \rightarrow_W G$, $F \rightarrow_W G$

Exclusive: $B \#_W C$

Parallel: $E ||_W F$

d

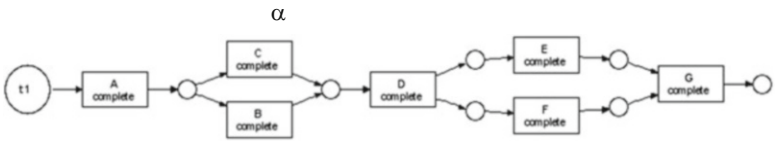


Fig. 7.5 Principles of the α-algorithm and example (using ProM 5.2). (**a**) Process execution log, (**b**) analysis of order relations over process log, (**c**) result of analysis step, and (**d**) resulting Petri net (using α-algorithm in ProM 5.2)

kind of technique, i.e., not based on frequency counting from the log, but based on evolutionary algorithms as optimization methods. For a detailed overview and discussion on existing process discovery techniques, we refer to [60].

Consider the log data provided in Fig. 7.5a, consisting of five cases (i.e., process instance executions), assuming that the occurrences of events connected to task executions in the log reflect their execution order. Note that for applying tools such as ProM,[8] this log has to be provided or transformed into a log format, e.g., MXML or XES (cf. Sect. 3.4.2).

The *α-algorithm* starts with analyzing the order between activity entries within each single log. For `Case1`, for example, activity entry `A` has occurred before execution of `B`, and hence, `A` is considered a predecessor of `B`. This relation between two activities is denoted by $A \longrightarrow_W B$.

[8] http://www.processmining.org/prom/start.

After analyzing each log separately, the relations derived from each log are aggregated across all available logs. For each pair of activities, this aggregation results in one of the three basic relations depicted in Fig. 7.5b, i.e., sequential order, parallel order, or exclusive order (e.g., for each instance, either `B` or `C`, but never together). The results are summarized in Fig. 7.5c:

- An example for a sequential order is $A \longrightarrow_W B$ as `A` was observed before `B` for all logs.
- An example for activities ordered in parallel is $E \,||_W\, F$ because `E` happened before `F` and `F` happened before `E` for some logs.
- An example for activities ordered exclusively is $B \,\#_W\, C$ as there is no log containing `B` and `C`.

The aggregated relations displayed in Fig. 7.5c are used to construct the resulting process model. In our example, the Petri net as depicted in Fig. 7.5d will result from the analysis (constructed using ProM 5.2[9]).

In summary, the α-algorithm finds basic relations among activities and constructs the resulting process model.

HEP Use Case: Application of α-algorithm

Figure 7.6 illustrates a more complex log example from the HEP case study. 74 process instances (cases) included in the log result in 4,018 events. The mined Petri net model looks quite complex. Such complex or unstructured models are referred to as "spaghetti models," whereas structured process models resulting from process discovery are denoted as "lasagna models" [60]. Details on the HEP project can be found on the homepage of the book:

www.businessintelligence-fundamentals.com

Intuitively, understanding and working with spaghetti models can be cumbersome and error-prone. The reasons for the occurrence of spaghetti models are manifold. For example, such models might result from the characteristics of the underlying process structures and the possible behavior of process actors during process execution (*process-related problems*). According to [17], factors of influence for spaghetti models can be as follows: log data contributed by parallel branches, infrequent traces, and log data contributed by individually modified instances.

The heuristic miner introduced in the sequel is able to deal with infrequent behavior and hence—at least to some extent—with modified instances. Note that it cannot distinguish between behaviors that are part of the process model, but only rarely occurring behavior, and behavior that has been added by an instance change. The phenomena deviating from the prescribed process model during the real-life execution is referred to as *concept drift* [15]. How to exploit information on process deviations and changes is discussed in Sect. 7.4.2.

[9] We use ProM 5.2 to present the principles of the α-algorithm, the heuristic miner, and the genetic miner. For later examples, we will use the more current version ProM 6.3.

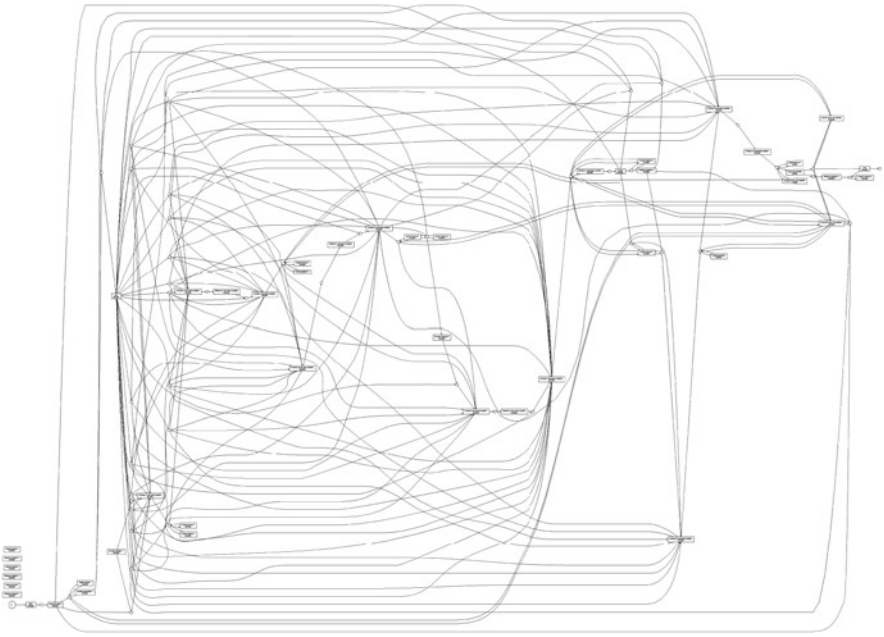


Fig. 7.6 α-Algorithm applied to HEP log example (using ProM 5.2)

Spaghetti models might also occur due to low log data quality and noise such as incorrect or missing data [16] (cf. Sect. 3.5.2).

Process-related and data-related problems can be tackled from either the algorithmic side or by providing *preprocessing* [8] and *post-processing* [15] strategies for the data or resulting process models, respectively.

In order to be more robust against noise, several process mining algorithms have been developed. One example mentioned before is the heuristic miner that infers the ordering relations between tasks based on their frequencies within the log. At first, it is counted how often activity a directly follows activity b in event log W (denoted by $a >_W b$). Then it is counted how often b is directly following a in event log W, i.e., $b >_W a$. If the number $|a >_W b|$ is much higher than the number $|b >_W a|$, it can be concluded that a also causally follows b [37]. The following formula taken from [37] reflects the above description:

$$a \Rightarrow_W b = \frac{|a >_W b| - |b >_W a|}{|a >_W b| + |b >_W a| + 1}. \tag{7.1}$$

Example 7.1 (Application of the Heuristic Miner) Consider the log given in Fig. 7.7. The result of applying the heuristic miner shows that for activities `A` and `B`, the relation $A >_W B$ can be found twice in the log. The opposite relation $B >_W C$ is not present, resulting in a frequency of $\frac{2}{2+1} = 0.67$. Relation

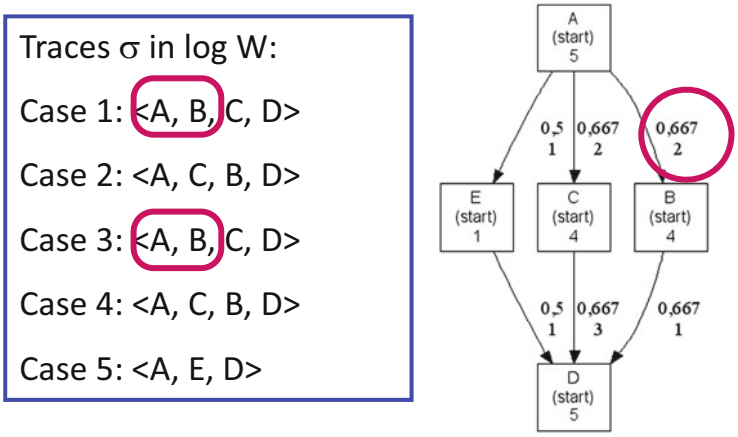


Fig. 7.7 Example log and result of applying heuristic miner (using ProM 5.2)

$B >_W C$, for example, is observed twice within the logs as well as the opposite relation $C >_W B$. Hence, the probability of a causal relation of $B >_W C$ turns out as $\frac{0}{1}$. Hence, `B` and `C` are evaluated as being parallel.

`HEP Use Case: Applying Heuristic Miner`

Recall the result of applying the α-algorithm to the HEP event log data as depicted in Fig. 7.6. This time, we discover the process model based on the heuristic miner (cf. Fig. 7.8). It can be seen that the model is less complex, i.e., it contains less paths and the flow direction is visible. Roughly, we could call the model in Fig. 7.6 a spaghetti model and the model in Fig. 7.8 a lasagna model. The reduction of the number of paths is caused by cutting infrequently occurring paths. Overall, we can state that in this case, application of the heuristics miner yields a more compact and understandable process model than the application of the α algorithm. Details on the HEP project can be found on the homepage of the book:

www.businessintelligence-fundamentals.com

Another process mining technique that is resilient toward noise is the *genetic miner* [5] which follows the basic principle of evolutionary algorithms [10]. Such algorithms serve as heuristic optimization techniques that are inspired by natural evolution processes. The basic problem setting is as follows: given an initial solution and an optimization goal (e.g., minimize or maximize a certain value), how can the initial solution evolve towards fulfilling the optimization goal? Translated into a process discovery problem, starting from a set of individuals, i.e., process models, we try to optimize them towards the optimal model, i.e., that model that best reflects the given set of event logs.

Doing so, different challenges have to be addressed. We will explain them in the following and then illustrate the different steps.

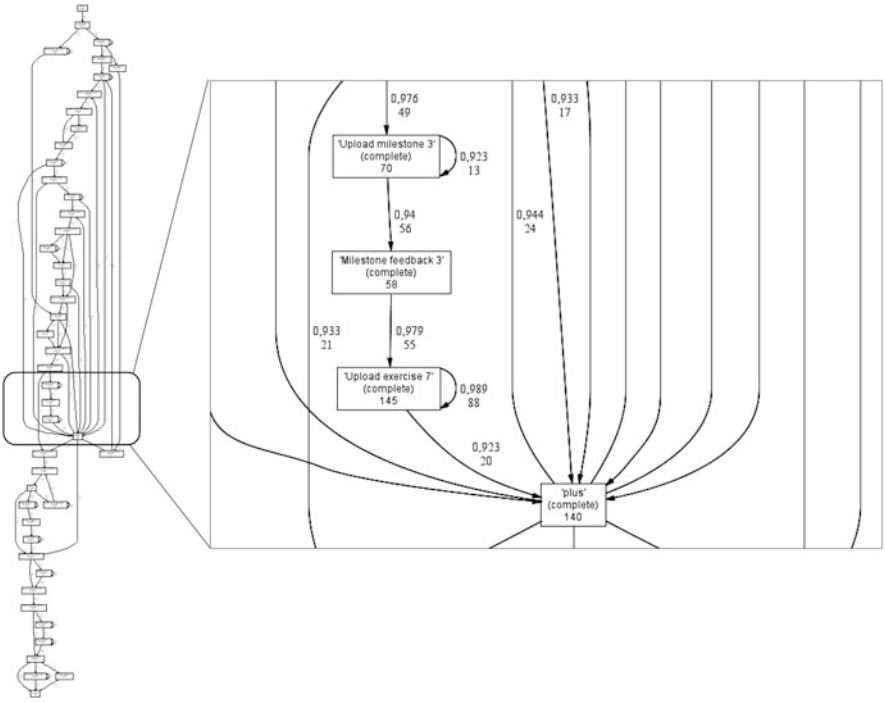


Fig. 7.8 Heuristic miner applied to HEP log example (using ProM 5.2)

Steps of the Genetic Miner

1. *Encoding individuals:* A first important consideration is how to encode these individuals for optimization, i.e., finding an adequate encoding as genotype for the phenotype of the individuals. The genetic miner encodes the individuals, i.e., process models, as *causal matrices*. A causal matrix for a process model contains the input and output functions, i.e., the sets of activity set that enable or are being enabled by the execution of the activity, respectively.
2. *Creating initial solution:* If the encoding is chosen, an initial population has to be generated, i.e., individuals have to be selected. In general, either an initial population is at hand or created. For the genetic miner, the initial population is generated by taking into consideration all activities present in the logs and imposing random causal relationships on them (e.g., 50 % chance that a causal relation is established between two activities). Heuristic extensions exist that take into consideration frequencies of paths contained within the logs.
3. *Fitness and selection of individuals:* An initial population at hand, it is now important to have a means to measure the quality of the individuals. For this, in general, a *fitness function* is defined that calculates the quality of an individual with respect to the optimization goal and based on the chosen encoding.

The genetic miner employs a fitness function that punishes for imprecise and incomplete process models.

The terms imprecise and incomplete are tightly connected with evaluating the results of process discovery in general as well as with the question of conformance checking. A model is incomplete if not all traces of the log can be replayed on this model. It is imprecise if it allows for producing additional traces that are not present in the log. We will explain the associated terms and techniques in detail in Sects. 7.4.3 and 7.6.

4. *Creating new offspring:* Based on a certain threshold and respecting the diversity of the solution (i.e., not always all the fittest individuals are selected in order to avoid local optima), those individuals are selected that create the next generation (offspring). In order to create offsprings, the two operations *mutation* and *crossover* can be used. Mutation works on one individual, whereas crossover combines the information of two individuals. For the genetic miner, mutation takes the causal matrix of an individual and adds new information to the causal matrix. This can be achieved by, for example, adding an activity from the underlying activity set to the input/output activity sets or by deleting an activity, respectively.
5. *Evaluating offspring:* Based on evaluating the offspring by using the fitness function, new parents for the next iteration are chosen, and so on, until a stop criterion is reached (e.g., the optimum—if known—is reached or a given threshold for the number of iterations is exceeded).

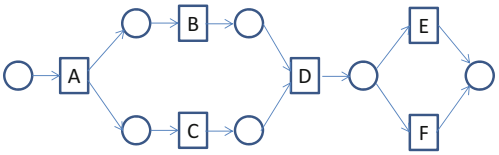
HEP Use Case: Application of Genetic Miner

In Fig. 7.9a, we can see a process model represented as a Petri net on the left side and its encoding as causal matrix on the right side. For activity `A`, for example, the set of activities that have been causally preceding `A` is empty. This means that `A` can start immediately. After completing `A`, two activities `B` and `C` follow causally. Hence, these two activities form the output set of `A`.

Figure 7.9b, c show the different settings for the genetic miner as discussed above as well as the result for mining the HEP log as already presented for the α-algorithm (cf. Fig. 7.6) and the heuristic miner (cf. Fig. 7.8). It can be seen that after 1,000 iterations, the individuals expose a very low fitness. This is also reflected by the resulting spaghetti model. Hence, comparing the results of alpha-algorithm, heuristic miner, and genetic miner shows that the heuristic miner yields the most compact result. This can be explained by the fact that the heuristic miner removes infrequent traces. Thus, we can conclude that the underlying process shows several variations or even exceptional situations. Details on the HEP project can be found on the homepage of the book:

www.businessintelligence-fundamentals.com

a



Activity x	I(x)	O(x)
A	{}	{{B,C}}
B	{{A}}	{{D}}
C	{{A}}	{{D}}
D	{{B,C}}	{{E},{F}}
E	{{D}}	{}
F	{{D}}	{}

b

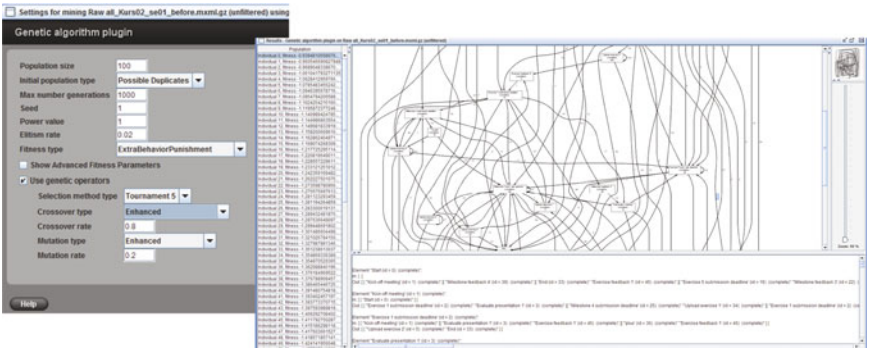


Fig. 7.9 Genetic miner basics and HEP use case (Net in c) (using ProM 5.2). (**a**) Process Model (Petri Net) and corresponding Causal Matrix, (**b**) settings for Genetic Miner (ProM 5.2), and (**c**) result for HEP example

As the genetic miner is capable of processing suboptimal solutions, i.e., process models, it is also capable to process noisy event logs.

7.4.2 Change Mining

In addition to discovering process models from event data, it can also be of high interest for users to learn about previously applied process changes, particularly when they are about to apply a process change themselves [26, 35]. In the medical domain, for example, it could be helpful to review what other specialists have done in a similar situation. Hence, *change mining* [12] has been developed as means to mine and learn about process changes that have been applied to process instances previously.

Before describing change mining in more detail, the topic of process change will be introduced in a nutshell. Process change is a well-established research topic and relevant for almost any application domain. Basically, one can distinguish between design time and runtime process flexibility. Design time flexibility is referred to

by, for example, the underspecification of parts of the process models and late modeling/binding techniques. Runtime flexibility refers to the ability of adapting running process instances during their execution. For a discussion on process change, see [22, 25].

Now we come back to change mining: change mining is not based on execution logs, but on change logs [28]. Change logs store information about the kind of change operation that has been applied. They use parameters such as time or change originator. Examples for change operations are `INSERT(S, X, A, B)` and `DELETE(S, Y)`. In the first case, an activity `X` is inserted into schema S between activities `A` and `B`. In the second case, activity `Y` is deleted from schema S. A survey on process change operations and patterns may be found in [34].

Consider the change logs τ_1 and τ_2 (cf. execution logs in Sect. 3.4.2):

τ_1 =< = INSERT(S, X, A, B), DELETE(S, Y), INSERT(Z, S, D, E)>
τ_2 =< = DELETE(S, Y), INSERT(S, X, A, B)>

Change mining basically applies the heuristic miner to change logs, resulting in *process change graphs*. In other words, instead of constructing the process execution graph, i.e., the process model, the change miner derives the process change graph, i.e., the model that includes all changes that have been applied at instance level in an aggregated manner.

`Logistics Use Case: Application of Change Mining`

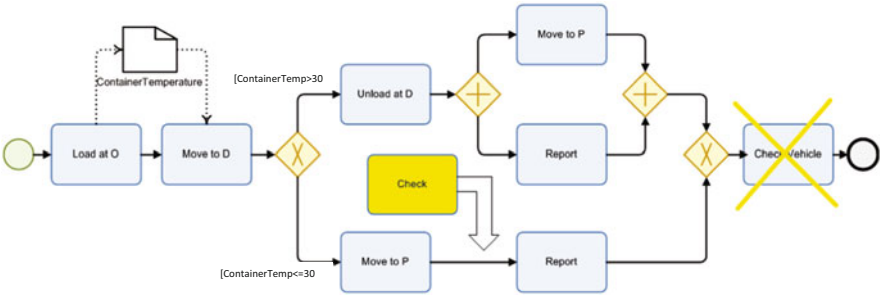
Figure 7.10a shows our example from the logistics domain on container transportation. Assume that several process instances have been executed based on this schema. Assume further that for several of these instances change operations have been applied, i.e., inserting the new activity `Check` between the activities `Move to P` and `Report` (`INSERT(S, Check, Move to P, Report)`) and deleting activity `Check Vehicle` (`DELETE(S, Check Vehicle)`). The information on the applied changes is collected for each instance and stored within associated change logs. Figure 7.10b depicts a change log entry for inserting activity `Check` for one of the instances.

Figure 7.11 illustrates the result of applying the change miner algorithm to the change logs resulting in an event-driven process chain. Within this description language, it can be easily seen that both changes were applied separately but also together. An implementation of change mining is available in ProM 5.2. Details on the container transportation use case can be found on the homepage of the book:

www.businessintelligence-fundamentals.com

In this simple example, the overview on applied changes and their relations can easily be kept. However, for more complex change settings as described in [12], the change mining results can provide a useful tool for users in order to analyze previously applied changes. This information can be taken as input for future change decisions.

a



b

```
<AuditTrailEntry>
  <Data>
    <Attribute name="CHANGE.postset">Report</Attribute>
    <Attribute name="CHANGE.type">INSERT</Attribute>
    <Attribute name="CHANGE.subject">Check</Attribute>
    <Attribute name="CHANGE.rationale">Additional Check
        required</Attribute>
    <Attribute name="CHANGE.preset">Move to P</Attribute>
  </Data>
  <WorkflowModelElement>INSERT.Check</WorkflowModelElement>
  <EventType>complete</EventType>
  <Originator>Dr. Smith</Originator>
</AuditTrailEntry>
```

Fig. 7.10 Container transportation process with changes and corresponding change log. (**a**) Container process with change operations and (**b**) log data entry

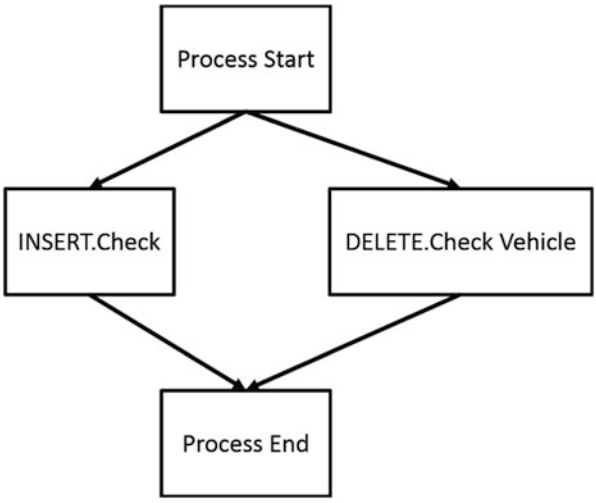


Fig. 7.11 Container transportation process: result of applying change mining

7.4.3 Conformance Checking

Process discovery works on a set of execution logs. Conformance checking requires both as input execution logs and a process model. As stated in [29, 60], the general goal of conformance checking is to identify commonalities and deviations between the real-world process (reflected by the logs) and the modeled process. Hence, conformance checking can be used for compliance checking and auditing (cf. Sect. 7.5). We can estimate conformance checking as supervised learning technique.

Assume the existence of some process model P represented by a Petri net and a set of event logs E. The degree to which P reflects E can be measured based on a conformance metrics that counts the number of tokens that remain in the process model (*remaining tokens*) and tokens that are missing for process execution along the model after having replayed the entire log data (*missing tokens*).[10]

Remaining tokens occur if the traces have been completely replayed and—except for the final state—at least one state is marked by a token. Missing tokens reflect how many tokens are missing for activating process activities that are in the log.

The formula that puts remaining and missing tokens into one number is as follows [29]:

$$f = 0.5 * \left(1 - \frac{\sum_i n_i * m_i}{\sum_i n_i * c_i}\right) + 0.5 * \left(1 - \frac{\sum_i n_i * r_i}{\sum_i n_i * p_i}\right) \tag{7.2}$$

where

- n_i is the number of activities in the process model
- m_i is the number of missing tokens
- c_i is the number of consumed tokens
- r_i is the number of remaining tokens
- p_i is the number of produced tokens

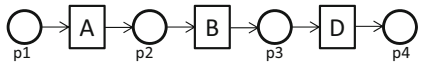
The following abstract example illustrates how the conformance metrics works:

Example 7.2 (Conformance Metrics Based on Abstract Petri Net Example) Consider the example depicted in Fig. 7.12. Given a log of 8 traces and the depicted Petri net model P, conformance metric f [29] determines a weighted measure between missing and remaining tokens for all logs weighted by the number of logs, and the number of tokens that have been consumed and produced according to the logs on the net. These consumed and produced tokens reflect the conformance of the model to the logs, whereas remaining and missing tokens reflect deviations which are "punished" within the conformance metrics f.

Intuitively, P conforms with 4 of the logs, i.e., <A, B, D>, whereas the other 4 logs, i.e., <A, C, D>, cannot be completely replayed on P. The corresponding

[10] Note the similarity between the conformance metrics and the fitness function for genetic mining as presented in Sect. 7.4.1.

Petri Net model P:



<A, B, D>
<A, B, D>
<A, B, D>
<A, B, D>
<A, C, D>
<A, C, D>
<A, C, D>
<A, C, D>

	<A, B, D>: i = 1	<A, C, D>: i = 2
Remaining tokens r_i	0	1
Missing tokens m_i	0	1
Produced tokens p_i	3	2
Consumed tokens c_i	3	2
Conformance:	$f=0.5*(1-\frac{3*0+3*1}{3*3+3*2}) + 0.5*(1-\frac{3*0+3*1}{3*3+3*2}) = 0.8$	

Fig. 7.12 Measuring conformance based on an artificial example

table shows the different numbers for the conformance analysis: for logs <A, B, D>, one token will remain on P, i.e., on place $p2$. The reason is that since B is not present in the log, it is not executed. Hence, it does not consume the token on $p2$. In turn, one token will be missing, i.e., the one which will trigger D. For logs <A, C, D>, two tokens are produced and consumed, respectively, by executing A and D. For the other 4 logs <A, B, D>, there are neither remaining nor missing tokens and 3 tokens are consumed and produced. By putting all the numbers into the formula, f turns out to be 80 %.

Petri net model P (cf. Fig. 7.12) represents only a part of the real-world behavior reflected in the logs. Hence, its fitness, i.e., its ability to reflect the log, turns out to be 80 %.

7.4.4 Summary: Process Mining

Process mining basically tackles two analysis questions: (a) based on a set of process execution logs, how does the underlying process model look like (process discovery) and (b) based on a set of process execution logs and a process model, how well does the model reflect the real-world behavior as reflected by the logs (conformance checking)? Process mining techniques have been proven to be relevant in many application domains. In [60], case studies for municipalities and the manufacturing domain are provided. In our EBMC^2 project, for example, the heuristic miner was applied to an initial set of the patient treatment processes. When discussing the results with domain experts, they observed several activities and process executions that aligned with the guideline. Though ten cases are too few to be representative, the experts stated that process discovery is useful as a *screening* tool [7].

Both algorithms and tool support for process mining have made significant progress during the last years. One remaining challenge is that data is often not available in a process-oriented structure and format. This demands for techniques

that enable the extraction, integration, and transformation of existing data into process logs. A relevant question for future research is how to integrate process mining techniques with further analysis techniques, e.g., data mining. This seems to be promising with respect to gaining more insights into the data and to be able to address more analysis questions. In Sect. 8, we will present existing combined techniques, for example, for mining organizational structures and discuss current limitations and open questions.

7.5 Business Process Compliance

Business process compliance has developed as of one of the key concerns for process-oriented applications nowadays through many application domains such as finance, security and privacy, health care, service flows, and internal controls. The general question of business process compliance is to check and ensure that business processes and workflows obey to the relevant constraints, rules, guidelines, and controls imposed on the business processes. For simplicity reasons, we will refer to constraints in the following.

7.5.1 Compliance Along the Process Life Cycle

Business compliance constitutes a challenge throughout the entire business process life cycle [17] ranging from compliant-by-design business processes and design time compliance checks to runtime monitoring approaches.

At design time, different analysis questions arise (cf. Fig. 7.13). One question is whether relevant constraints are entirely considered within the process model or whether there is a co-existence between process models and constraints. In the first case, one can distinguish between imperative and declarative process models. For imperative process models, constraints can be either directly captured within the model design or specified by annotations.

Considering constraints within the models is particularly supported by declarative process modeling notations. Compared to the imperative notions used throughout this book, declarative process models consist of constraints which express themselves what can be done and what cannot be done instead of imposing a strict process execution. If process models are specified in a declarative manner, compliance constraints can be added. Then compliance checking means to identify consistency between the constraints.

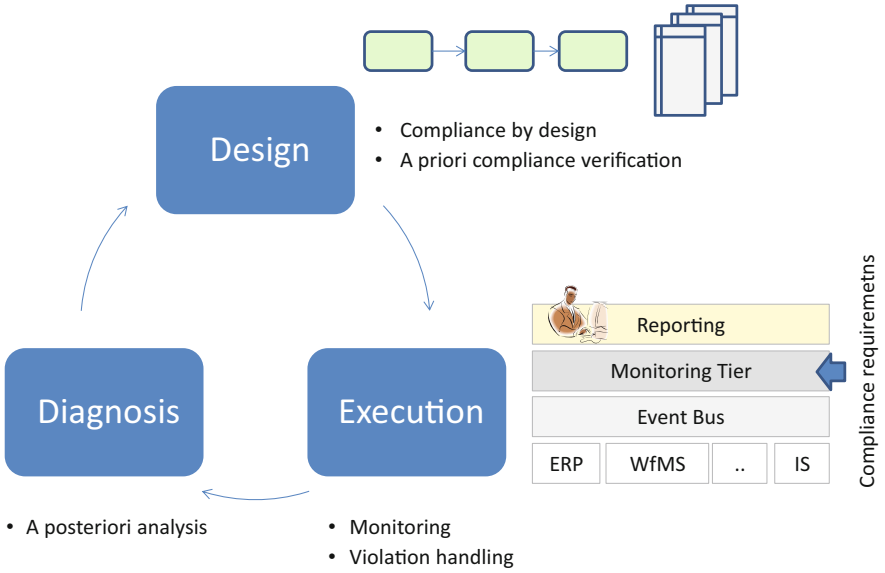


Fig. 7.13 Overview on compliance approaches along process life cycle (following [18])

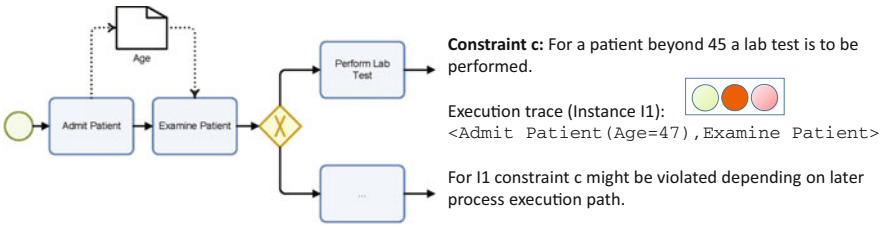


Fig. 7.14 Medical example compliance constraint

At process execution time, it often becomes necessary to monitor the adherence of running processes with imposed compliance constraints due to two aspects: (a) certain aspects of constraints that cannot be checked during design time, e.g., data or time, and (b) sometimes process models are not available. As indicated in Fig. 7.13, process execution information might be available as events associated with process activity executions and stemming from different underlying information systems. In this case, compliance constraints have to be verified on the fly. For an overview and comparison of existing approaches for compliance monitoring, see [18].

Special challenges arise not only in verifying compliance constraints but also in reporting back on violations. In particular, a `true/false` answer might not be sufficient in all cases, but reasons for compliance violations or even feedback for healing violations are required. The example in Fig. 7.14 shows a fragment of a medical treatment process and an associated constraint c. Based on the execution trace for a certain instance $I1$, we can conclude that c might be possibly violated in

case the "wrong" path of the XOR branching is chosen. Hence, it could be favorable to suggest the "right" path to the user as proactive handling of a possible violation.

A diagnosis can also comprise compliance-related analysis. In the security domain, for example, so-called a posteriori analysis techniques can be applied to detect security violations during process executions [2]. One set of techniques applied at a posteriori time are process mining approaches such as process discovery, process conformance, or LTL checking.

7.5.2 Summary: Compliance Checking

Business process compliance constitutes one of the most challenging questions for enterprises nowadays, and tremendous effort has been spent on providing system-based support for compliance integration and checking in process-aware information systems. The techniques have well matured. However, the challenge is to acquire, model, and formalize relevant compliance constraints over business process for later analysis. Another challenge is the interaction with users in case of compliance violations. In most cases, the pure indication of such a violation is not enough to enable the user to deal with the violation. Hence, advanced strategies for handling compliance violations in a user-friendly way are required.

7.6 Evaluation and Assessment

How can we evaluate and assess the results or process analysis?

7.6.1 Process Mining

The quality of a discovered process model after mining it from an event log depends on different dimensions. According to [30], these dimensions are *fitness*, *precision*, *generalization*, and *structure*. Fitness and precision have been already discussed along conformance checking (cf. Sect. 7.4.3): fitness refers to how much of the behavior described in the logs is reflected by the discovered model. The concern of precision is that the behavior as described by the logs should be reflected as exactly as possible within the model, i.e., no additional behavior should be producible. Both dimensions, fitness and precision, are captured within the conformance metrics by punishing for behavior that is not reflected and behavior that is additionally reflected.

The evaluation framework in [30] additionally introduces generalization as dimension. Generalization constitutes some kind of counterpart to precision. It aims at avoiding models that can only reflect the behavior of the logs, but no more behavior. Of course, generalization and precision are to be balanced, i.e., the right level of generalization must be determined.

Finally, structure evaluates the process model in terms of complexity. It is possible, for example, to have two process models that are execution equivalent, i.e., bear the same behavior, but are of different structure. One model might contain, for example, duplicate or silent activities[11] for structuring reasons. Then, a metric evaluating the structure would rate the quality of the more compact model higher than the one of more complex structure.

7.6.2 Compliance Checking

Evaluation in the context of process compliance checking might refer to two aspects, i.e., efficiency and quality of the feedback. Here, efficiency means how long compliance checks take when facing complex process models and a multitude of compliance constraints. Efficiency in the sense as discussed here is mostly relevant for automated compliance checks by a tool or system. Nowadays, compliance is often checked manually by, e.g., auditors. In such settings, it is more difficult to evaluate efficiency.

The quality of feedback is a crucial aspect as the strategy of how to deal with compliance violation depends on the expressiveness of this feedback. At least, compliance checking solutions should precisely pinpoint the source for the violation. Even better, they can offer possible solution strategies when the violation has already happened or proactively warn users in case of potential violations.

Overall, process analysis techniques are of high interest to companies as they provide means to optimize business processes, discover process models, compare different process models, and evaluate the compliance of business processes with a set of relevant constraints.

7.7 Conclusion and Lessons Learned

According to our experience and as it is shown by the case studies provided in this book, the interest in process analysis techniques spreads over all kinds of application domains such as health care, manufacturing, or logistics. The adoption of process analysis techniques would be fostered by providing systematic guidelines or methods of how to apply the different techniques. In addition, the interpretation

[11] Silent activities are not connected with any action and hence do not produce any log entry.

of the results often requires an advanced understanding of the techniques, but also the domain of interest. In times of big data being a hype, the trend of big process data will raise additional tasks such as the proper visualization of analysis results.

7.8 Recommended Reading

Introductions to the areas of business process management and process-aware information systems are provided in Weske (2007) and Dumas (2013). Dumas (2013), moreover, discusses process analysis techniques. van der Aalst (2011) provides introductory material as well as details on process mining and analysis techniques, applications, and related areas.

- Weske, M (2012) Business process management—concepts, languages, architectures. Springer, Heidelberg, 2nd edition
- Dumas M, La Rosa M, Mendling J, Reijers HA (2013) Fundamentals of business process management. Springer, Heidelberg
- van der Aalst WMP (2011) Process mining—discovery, conformance and enhancement of business processes. Springer, Heidelberg

References

1. Accorsi R, Stocker T (2013) SecSy: synthesizing smart process event logs. In: Jung R, Reichert M (eds) EMISA'13: enterprise modelling and information systems architectures: proceedings of the 5th international workshop on enterprise modelling and information systems architectures workshop. Lecture notes in informatics, vol 222, pp 71–84
2. Accorsi R, Stocker T, Müller G (2013) On the exploitation of process mining for security audits: the process discovery case. In: Shin SY, Maldonado JC (eds) SAC'13: annual ACM symposium on applied computing. ACM, New York, pp 1462–1468
3. Becker J, Bergener P, Breuker D, Räckers M (2012) An empirical assessment of the usefulness of weakness patterns in business process redesign. In: ECIS 2012: 20th European conference on information systems
4. de Medeiros AKA, van der Aalst WMP, Weijters AJMM (2003) Workflow mining: current status and future directions. In: Meersman R, Zahir T, Schmidt DC (eds) OTM'03: On the move to meaningful internet systems 2003: CoopIS, DOA, and ODBASE. Lecture notes in computer science, vol 2888. Springer, Heidelberg, pp 389–406
5. de Medeiros AKA, Weijters AJMM, van der Aalst WMP (2007) Genetic process mining: an experimental evaluation. Data Min Knowl Discov 14(2):245–304
6. Dumas M, La Rosa M, Mendling J, Reijers HA (2013) Fundamentals of business process management. Springer, Heidelberg
7. Dunkl R, Binder M, Dorda W, Fröschl KA, Gall W, Grossmann W, Harmankaya K, Hronsky M, Rinderle-Ma S, Rinner C, Weber S (2012) On analyzing process compliance in skin cancer treatment: an experience report from the evidence-based medical compliance cluster (EBMC2). In: Ralyte J, Franch X, Brinkkemper S, Wrycza S (eds) CaISE'12: International conference on advanced information systems engineering. Lecture notes in computer science, vol 7328. Springer, Heidelberg, pp 398–413

8. Fahland D, van der Aalst WMP (2012) Simplifying discovered process models in a controlled manner. Inf Syst 38(4):585–605
9. Gabler Wirtschaftslexikon, Springer. http://wirtschaftslexikon.gabler.de/Definition/business-process-reengineering.html. Accessed 28 May 2014
10. Goldberg DE, Holland JH (1988) Genetic algorithms and machine learning. Mach Learn 3(2):95–99
11. Grigori D, Casati F, Dayal U, Shan M-C (2001) Improving business process quality through exception understanding, prediction, and prevention. In: Apers PMG, Atzeni P, Ceri S, Paraboschi S, Ramamohanarao K, Snodgrass RT (eds) VLDB'01: International conference on very large daTa Bases. Morgan Kaufmann, San Francisco, pp 159–168
12. Günther C, Rinderle-Ma S, Reichert M, van der Aalst WMP, Recker J (2008) Using process mining to learn from process changes in evolutionary systems. Int J Bus Process Integr Manag 3(1):61–78
13. Hildebrandt T, Rinderle-Ma S (2013) Toward a sonification concept for business process monitoring. In: ICAD'13: International conference on auditory display
14. Hildebrandt T, Kriglstein S, Rinderle-Ma S (2012) Beyond visualization: on using sonification methods to make business processes more accessible to users. In: ICAD'12: International conference on auditory display
15. Jagadeesh Chandra Bose RP, van der Aalst WMP, Zliobaite I, Pechenizkiy M (2011) Handling concept drift in process mining. In Mouratidi H, Rolland C (eds) CaISE'11: International conference advanced information systems engineering. Lecture notes in computer science, vol 6741, pp 391–405. Springer, Heidelberg
16. Jagadeesh Chandra Bose RP, Mans RS, van der Aalst WMP (2013) Wanna improve process mining results? It's high time we consider data quality issues seriously. BPMcenter.org
17. Ly LT, Rinderle-Ma S, Göser K, Dadam P (2012) On enabling integrated process compliance with semantic constraints in process management systems—requirements, challenges, solutions. Inf Syst Front 14(2):195–219
18. Ly LT, Maggi FM, Montali M, Rinderle-Ma S, van der Aalst WMP (2013) A framework for the systematic comparison and evaluation of compliance monitoring approaches. In: Gasevic D, Hatala M, Motahari Nezhad HR, Reichert M (eds) EDOC'13: International enterprise distributed object computing conference. IEEE, Los Alamitos, California, Washington, Tokyo, pp 7–16
19. Mansar SL, Reijers HA (2005) Best practices in business process redesign: validation of a redesign framework. Comput Ind 56(5):457–471
20. Mansar SL, Reijers HA (2007) Best practices in business process redesign: use and impact. Bus Process Manag J 13(2):193–213
21. Pflug J, Rinderle-Ma S (2013) Dynamic instance queuing in process-aware information systems. In: Shin SY, Carlos Maldonado, C (eds) SAC'13: annual ACM symposium on applied computing, enterprise engineering track. ACM, New York, pp 1426–1433
22. Reichert M, Weber B (2012) Enabling flexibility in process-aware information systems—challenges, methods, technologies. Springer, Heidelberg
23. Reichert M, Rinderle-Ma S, Dadam P (2009) Flexibility in process-aware information systems. In: Jensen K, van der Aalst WMP (eds) Transactions on Petri nets and other models of concurrency. Lecture notes in computer science, vol 5460, Springer, Heidelberg, pp 115–135
24. Reijers HA, Liman Mansar SL (2005) Best practices in business process redesign: an overview and qualitative evaluation of successful redesign heuristics. Omega 33(4):283–306
25. Rinderle S, Reichert M, Dadam P (2004) Correctness criteria for dynamic changes in workflow systems—a survey. Data Knowl Eng 50(1):9–34
26. Rinderle S, Weber B, Reichert M, Wild W (2005) Integrating process learning and process evolution—a semantics based approach. In: van der Aalst WMP, Benatallah B, Casati F, Curbera F (eds) BPM'05: International conference on business process management. Lecture notes in computer science, vol 3649. Springer, Heidelberg, pp 252–267

27. Rinderle S, Bassil S, Reichert M (2006) A Framework for semantic recovery strategies in case of process activity failures. In: Manolopoulos Y, Filipe J, Constantopoulos P, Cordeiro J (eds) ICEIS'06: International conference on enterprise information systems, pp 136–143
28. Rinderle S, Reichert M, Jurisch M, Kreher U (2006) On representing, purging, and utilizing change logs in process management systems. In: Dustdar S, Fiadeiro SL, Sheth AP (eds) BPM'06: International conference on business process management. Lecture notes in computer science, vol 4102. Springer, Heidelberg, pp 241–256
29. Rozinat A, van der Aalst WMP (2008) Conformance checking of processes based on monitoring real behavior. Inf Syst 33(1):64–95
30. Rozinat A, de Medeiros AKA, Günther C, Weijters AJMM, van der Aalst WMP (2008) The need for a process mining evaluation framework in research and practice. In: ter Hofstede A, Benatallah B, Paik H-Y (eds) Business process management workshops. Lecture notes in computer science, vol 4928. Springer, Heidelberg, pp 84–89
31. van der Aalst WMP, Voorhoeve M (2014) Business process simulation. Lecture notes 2II75. http://wwwis.win.tue.nl/~mvoorhoe/sim/ln2II75.pdf?origin=publication_detail. Accessed 24 June 2014
32. van der Aalst WMP, Nakatumba J, Rozinat A, Russell N (2010) Business process simulation. Handb Bus Process Manag 1:313–338
33. van der Aalst WMP, Pesic M, Song M (2010) Beyond process mining: from the past to present and future. In: Pernici B (ed) CaISE'10: International conference on advanced information systems engineering. Lecture notes in computer science, vol 6051. Springer, Heidelberg, pp 38–52
34. Weber B, Reichert M, Rinderle-Ma S (2008) Change patterns and change support features—enhancing flexibility in process-aware information systems. Data Knowl Eng 66(3):438–466
35. Weber B, Reichert M, Rinderle-Ma S, Wild W (2009) Providing integrated life cycle support in process-aware information systems. Int J Coop Inf Syst 18(1):115–165
36. Weijters AJMM, Ribeiro JTS (2011) Flexible heuristics miner (FHM). In: CIDM'11: IEEE symposium on computational intelligence and data mining. IEEE, Los Alamitos, California, Washington, Tokyo, pp 310–317
37. Weijters AJMM, van der Aalst WMP, de Medeiros AKA (2006) Process mining with the heuristics miner-algorithm. Technische Universiteit Eindhoven, Technical Report WP 166
38. Weske M (2007) Business process management—concepts, languages, architectures, 2nd edn. Springer, Heidelberg