Decision Discovery in Business Processes

Massimiliano de Leoni*, Felix Mannhardt†

1 Introduction

The main focus of automatic process discovery has traditionally been in the realm of process mining on the control-flow perspective (cf. the chapter of Automatic Process Discovery). As an example, the recent advances have led to discovery algorithms that can discover models such as that in Figure 1. This model only shows the control-flow, namely the ordering with which activities can be executed in the process.

Event logs are typically of such a form as in Figure 2. Automatic process discovery techniques only focus on the first two columns of an event log, the case identifier (e.g., the customer id) and the activity name, thus overlooking a lot of insightful information pertaining other perspectives, such as the organization perspective, the decision perspective (a.k.a. the data or case perspective) and the time perspective [24].

The control-flow perspective is certainly of high importance as it can be considered as the process backbone; however, many other perspectives should also be considered to ensure that the model is sufficiently accurate. This chapter will focus the attention on the decision perspective and will not discuss the organization and time perspectives. Readers are referred to [24] as initial pointer to explore the research results regarding the latter two perspectives.

The decision perspective focuses on how the routing of process-instance executions is affected by the characteristics of the specific process instance, such as the amount requested for a loan, and by the outcomes of previous execution steps, e.g. the verification result. The representation of this decision perspective on a process in an integrated model or as separate tables is nowadays gaining momentum. This is also testified by the recent introduction and refinement of Decision Model and Notation (DMN), which is a standard published by the Object Management Group to describe and model the decision perspective [1].

The simplest representation of the decision perspective is to attach decision rules to XOR and OR splits, the decision points of a process model. The rules explaining the choices are driven by additional data associated with the process, which is generated by the previous process' steps. In the remainder, this additional data is abstracted as process attributes, each of which is name-value pair. Note that not all decision points

*Dr. de Leoni, Eindhoven University of Technology, The Netherlands m.d.leoni@tue.nl
†Dr. Mannhardt, SINTEF Technology and Society, Norway felix.mannhardt@sintef.no

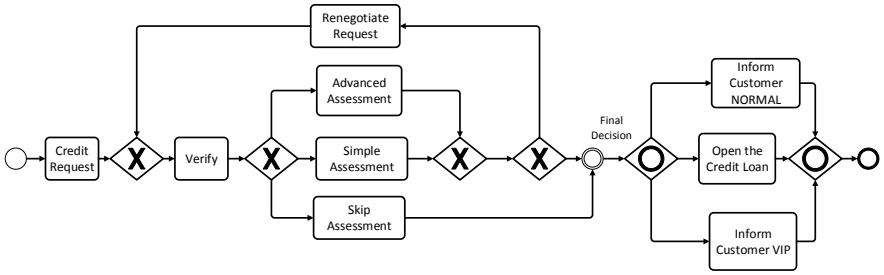

Figure 1: A model that shows only the control-flow, i.e. the ordering of execution of activities.

are driven by rules: the process attributes are not able to determine which branch is going to be followed (cf. the XOR split involving *Renegotiate Request*).

Historically, the discovery of the decision perspective is called *Decision Mining*, a name that was introduced by the seminal work of Rozinat et al. [22]. In this work, the mined decisions at the decision points are mutually exclusive: when a process instance reaches a decision point, one and exactly one branch is enabled. Rozinat et al. leverages on Petri nets but the same idea can trivially be moved to BPMN and other process modelling notations. In this work, decision-mining problems are transformed into classification problems: techniques such as decision-tree learning can be leveraged off. Decision trees are learnt from an observation-instance table. For each decision point in the process, the subset of the events that refers to the activities that follow the decision point is extracted. Each observation instance consists of a dependent variable, a.k.a. response variable, and a set of independent variables, a.k.a. predictor variables, that are used for prediction. The dependent variable is the activity of the event following and (a subset of) the event (i.e. process) attributes are used as independent variables.[1]

2 Decision Mining as a Classification Problem

The Introduction has illustrated that the problem of decision mining can be translated into a classification problem. Several classification techniques exist; however, one needs to focus on those techniques that return human-readable decision rules in form of, e.g., formulas. This motivates why most of the decision-mining techniques, such as the seminar work [22] and others [3, 9, 4], aim to learn decision trees and use them to build a set of decision rules. In addition to provide human-readable rules, decision trees have the advantage of creating rules that are clearly highlighting the attributes whose values affect the decisions. These works typically leverages on the C4.5 algorithm for decision-tree learning, which is capable to deal with noise and missing values. In the context of decision mining, noise indicates that in a fraction of situations the wrong (or

[1]This chapter denotes variables as the features used in observation instances to learn a (decision-tree) classifier. Attributes are conversely associated with processes and events and indicate the data points manipulated while executing a process instance.

Customer id	Activity Name	Timestamp	Amount	Resource	Verification	Interest	Decision	Applicant	VIP
HT25	Credit Request	1-12-2012 9:00AM	3000	John	-	-	-	Max	1
HT25	Verify	3-12-2012 1:00PM	3000	Pete	OK	-	-	Max	1
HT25	Simple Assessment	7-12-2012 11:00AM	3000	Sue	OK	599	NO	Max	1
HT25	Inform Customer VIP	7-12-2012 3:00PM	3000	Mike	OK	599	NO	Max	1
HX65	Credit Request	3-12-2012 9:00AM	5000	John	-	-	-	Felix	0
HX65	Verify	5-12-2012 1:00PM	5000	Mike	NO	-	-	Felix	0
HX66	Skip Assessment	5-12-2012 2:00PM	5000	Mike	NO	-	-	Felix	0
HX65	Inform Customer NORMAL	8-12-2012 1:00PM	5000	Pete	NO	-	-	Felix	0
EA49	Credit Request	1-12-2012 9:00AM	6000	John	-	-	-	Sara	1
EA49	Verify	3-12-2012 1:00PM	6000	Pete	OK	-	-	Sara	1
EA49	Advanced Assessment	5-12-2012 1:00PM	6000	Sue	OK	1020	OK	Sara	1
EA49	Inform Customer VIP	8-12-2012 5:00PM	6000	Ellen	OK	1020	OK	Sara	1
EA49	Open the Credit Loan	8-12-2012 6:00PM	6000	Ellen	OK	1020	OK	Sara	1
HX12	Credit Request	10-12-2012 9:00AM	6000	John	-	-	-	Michael	1
HX12	Verify	13-12-2012 1:00PM	6000	Pete	OK	-	-	Michael	1
HX12	Advanced Assessment	15-12-2012 1:00PM	6000	Sue	OK	1020	NOK	Michael	1
HX12	Renegotiate Request	16-12-2012 3:00PM	6000	Sue	OK	1020	NOK	Michael	1
...	...	...	...	...	...	...	...	...	...

Figure 2: An excerpt of an event log. Each row is an event referring to an activity that happened at a given timestamp. Several process attributes are written and updated by one or more events. The alternation of two color tones is used to group events referring to the same process instance.

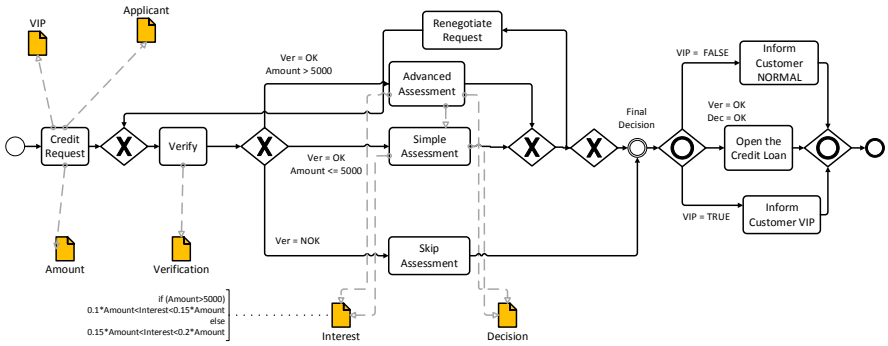

Figure 3: A model that extends what is depicted in Figure 1. The decision points are annotated with rules enabling the branches.

unusual) decision is made; missing values indicate that, due to logging errors, certain assignments of values to attributes are not recorded or, also, process participants forget to update the value of given attributes. Association-rule discovery techniques might be a possibility but they tend to return a large number of rules that not necessarily have a discriminatory function on the decision, either.

Let us consider the decision point in Figure 4, part of the model in Figure 1. Clearly, the rules are initially not present and the aim is to discover them. From the event log, every event that refers to the activities at the decision point is retained from the event log. Considering the event log in Figure 2, this results in a observation-instance table such as in Figure 5. The column *Activity Name* refers to the dependent variable and the other columns are the independent variables, categorical or numerical, that are used to predict the dependent variable. The application of decision-tree learning techniques

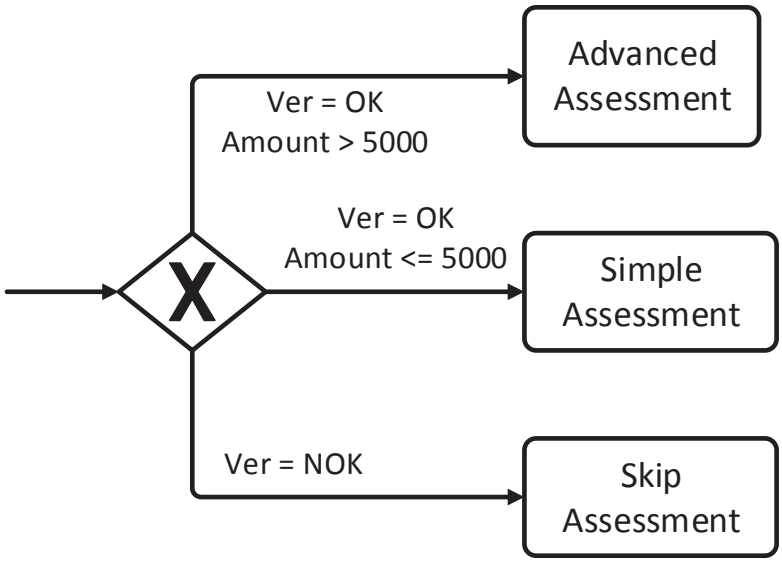

Figure 4: One of the decision points. The aim is to start from the control flow and discover the rules.

will produce a tree similar to that in Figure 6.

Decision trees classify instances by sorting them down in a tree from the root to some leaf node. Each non-leaf node specifies a test of some variable $x_1, \ldots, x_n$ (in decision mining, the process attributes) and each branch descending from that node corresponds to a range of possible values for this variable. In general, a decision tree represents a disjunction of conjunctions of expressions: each path from the tree root to a leaf corresponds to an expression that is, in fact, a conjunction of variable tests. Each leaf node is associated with one of the possible values of the dependent variables, namely with one of process activities that follow an XOR split: if an expression e is associated with a path to a leaf node a, every input tuple for which e evaluates to true is expected to return a as output. In decision mining, this means that, every time a decision point is reached and the process attributes $x_1, \ldots, x_n$ take on values that make e evaluate true, the process execution is expected to continue with activity a. In the example tree in Figure 6, each leaf refers to a different activity. However, multiple leaves can be associated with the same activity a. In that case, each path leading to a would correspond to a different expression. Together all these expressions combined with disjunction (i.e., a logical OR operator) constitute an expression that predicts when the process continues with activity a. As discussed in [24, pp. 294-296], this procedure is extensible to OR-splits, where multiple branches may be activated.

Customer id	Activity Name	Timestamp	Amount	Resource	Verification	Interest
HT25	Simple Assessment	7-12-2012 11:00AM	3000	Sue	OK	599
HX66	Skip Assessment	5-12-2012 2:00PM	5000	Mike	NO	-
EA49	Advanced Assessment	5-12-2012 1:00PM	6000	Sue	OK	1020
HX12	Advanced Assessment	15-12-2012 1:00PM	6000	Sue	OK	1020
...	...	...	...	...	...	...

Figure 5: An excerpt of the observation instances that are extracted from the event log in Figure 2 to discover the rules at the decision point in Figure 4.

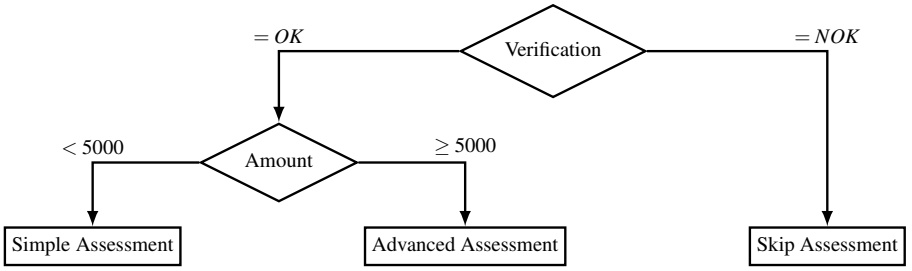

Figure 6: A possible decision tree discovered based on the observation instances of which an excerpt is in Figure 5.

3 Extension of the basic technique

This basic technique has been extended to deal with additional cases. The first extension is related to the situations when the executions are not always compliant with the control-flow model and/or the model contains invisible steps. The second extension is applicable when the rules at decision points are not mutually exclusive, such as when some information that drives the decisions is missing or the process behavior is simply not fully deterministic. These extensions are discussed in detail below.

There are other valuable extensions, which cannot be elaborated on in detail for the sake of space. In [4, 10], authors extend the basic algorithm to be able to discover the dependencies between process-attributes updates and leverage on invariant miners. For example, the values assignable to an attribute *Risk* are a function of the values taken on by attributes *Amount* and *Premium*. While the basic BPMN modelling notation is only able to represent path-routing decisions, BPMN can be complemented with DMN where data-related decisions can be represented. As discussed in [3], DMN is especially relevant when models are of significant size. By complementing BPMN with DMN, the decisions are modelled outside the control-flow model. The paper argues that this separation of concerns would increase the readability and the maintainability of these models. Here, maintainability of a model is intended as the easiness of extending, adjusting and updating a model when the internal protocols or the external rules and regulations change.

Customer id	Activity Name	Timestamp	Amount	Resource	Verification
TH25	Credit Request	1-11-2012 9:00AM	4000	John	-
TH25	Simple Assessment	7-11-2012 11:00AM	4000	Sue	-
TH25	Inform Customer VIP	7-11-2012 3:00PM	4000	Mike	-
TH26	Credit Request	1-1-2013 19:00AM	4100	Sue	-
TH26	Simple Assessment	1-2-2013 11:00AM	4100	John	-
TH26	Verify	3-2-2013 15:00PM	4100	Mike	NOK
TH26	Inform Customer NORMAL	17-2-2012 3:00PM	4100	Pete	NOK
...	...	...	...	...	...

Figure 7: Two non-compliant traces for the model in Figure 1: They break the assumption of the basic decision-tree algorithm.

3.0.1 Non-compliance and Invisible Steps.

The first assumption of the basic algorithm is that, when discovering the rules of a decision point, every activity is always executed in the right moment according to the dictation of the control-flow model, e.g. the model in Figure 1. However, due to logging errors and non-compliances, sometimes executions do not comply with the control-flow model. As an example, consider trace for the customer with id *TH25* in Figure 7. This trace is clearly not compliant with the model in Figure 1: activity *Credit Request* is immediately followed by *Simple Assessment*, i.e. activity *Verify* did not occur (or its execution was not logged). This means that the value of attribute *Verification* is missing. Several decision-tree learning algorithms, such as C4.5, are very good at dealing with missing values. However, it is necessary to detect that a value is missing due to non-compliancies. As an example, the execution for the customer with id *TH25*, the decision of executing *Simple Assessment* was made without considering the value of attribute *Verification*; in fact, the value of that attribute was undefined because the execution of activity *Verification* never took place. If such cases as this are not detected, one mistakenly could use an old value. For instance, attribute *Verification* is updated multiple times via multiple executions of activity *Verification*, which is executed once for each loan's renegotiation. If one renegotiation mistakenly skipped the verification, decision mining might use the previous value, updated during the previous renegotiation's loop. The basic algorithm also suffers of the problem that one should ignore updates of attributes that result from the non-compliant execution of activities. As an example, consider trace for the customer with id *TH26* in Figure 7: activity *Verify* occurs after *Simple Assessment*, which violates the constraints dictated by the model in Figure 1. As result of the execution of this activity, attribute *Verification* takes on value *NOK*; that is too late because a simple assessment has already been conducted. It is clear that, in this case, activity *Simple Assessment* should not have occurred.

To detect missing values and illegal attribute updates, the work report in [9] integrates the basic technique with techniques for log-model alignment. An alignment between a recorded process execution (log trace) and a process model is a pairwise matching between activities recorded in the log and activities allowed by the model.

Table 1 illustrates the alignments of the model in Figure 1 with the traces in Figure 7. Activity names are abbreviated with the first letter of each name word. Abstracting from the attribute updates and ignoring $\gg$'s, for each log trace, the log row

Customer with id *TH25*					
Log:	*CR*	≫	*SA*	*ICV*
Process:	*CrR*	*V*	*SA*	*ICV*

Customer with id *TH26*					
Log:	*CR*	*SA*	*V*	≫	*ICN*
Process:	*CR*	≫	*V*	*SA*	*ICN*

Table 1: Alignments of the traces in Fig. 7 wrt. the model in Fig. 1

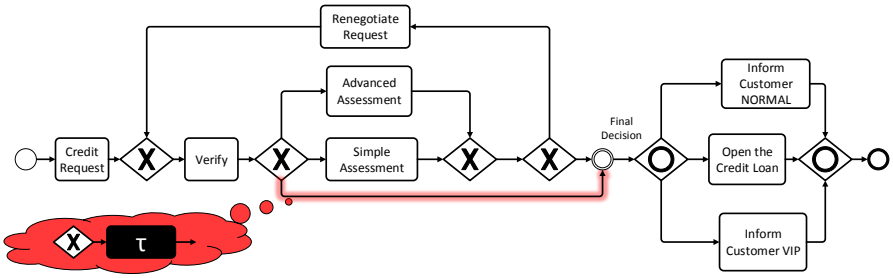

Figure 8: A variation of the model in Figure 1: activity *Skip Assessment* is removed. Ideally, it can be replaced by a silent activity τ, whose execution is not recorded in any trace of the event log.

correspond to the log trace and the process row represents the execution allowed by the process model that is the closest to the log traces in terms of number of necessary changes. The yellow and green columns refer to log and model moves, respectively. For further information, readers are referred to [2, 18]. The events for log moves will be ignored for decision mining. Conversely, activities related to model moves are introduced and will be considered. Attributes that are supposedly updated as result of the activity executions take on `null`, a special value to indicate that the value is missing and should be treated as such by the decision-tree learning algorithm. An alternative to using alignments could be to just remove the traces referred to executions not compliant with the control-flow model. However, situations in which most of traces are almost compliant and few are fully compliant would cause the rules to be mined on the basis of insufficient observation instances. If the missing events refer to activities that do not update attributes relevant for a certain decision point, there would be no reason to discard the corresponding traces when discovering the rules for such a decision point.

In addition to dealing with non-compliance, [9] allows for discovering decision rules for models such as in Figure 8. In this case, the explicit activity *Skip Assessment*, which indicates that no activity is performed in case of negative verification but, conversely, the execution directly moves to the negative final decision, is not present and recorded in the respective traces. In such a case, no activity can be explicitly identified to indicate that the assessment does not take place. To overcome this, the approach

in [9] would assume that there is a silent activity τ, as illustrated by the cloud in the figure. Using alignments, silent activity τ would be included in the analysis. When an execution of τ is necessary (e.g., when the assessment is skipped), the alignment would automatically add a model move for τ and, as mentioned above, model moves are always considered when mining the decision rules. Of course, the step τ is always involved in model moves since they are never recorded in the event log, being indeed silent.

3.0.2 Overlapping rules.

Existing decision mining techniques for exclusive choices rely on the strong assumption that the rules attached to the alternative activities of an exclusive choice need to be mutually exclusive. However, business rules are often non-deterministic and this *"cannot be solved until the business rule is instantiated in a particular situation"* [20] This ambiguity can occur due to conflicting rules or missing contextual information. For example, decisions taken by process workers may depend on contextual factors, which are not encoded in the system and, thus, not available in event logs [7]. In [17], a technique is proposed that discovers overlapping rules in those cases that the underlying observations are characterized better by such rules. The technique is able to deliberately trade the precision of mutually-exclusive rules, i.e., only one alternative is possible, against fitness, i.e., the overlapping rules that are less often violated. In short, as in the basic algorithm, the technique builds an initial decision tree based on observations from the event log and, for each activity a, a decision rule $rule1(a)$ is found (possibly, $rule1(a) = \mathsf{true}$). Then, for each decision tree leaf, the wrongly classified instances are used to learn a new decision tree leading to new rules, such as $rule2(a)$. These new rules are used in disjunction with the initial rules yielding, e.g., overlapping rules of the form $rule1(a) \vee rule2(a)$.

It is worth highlighting here that overlapping rules are appropriate in contexts where information is missing and/or rules are intrinsically not fully deterministic. However, in other contexts, overlapping rules are just the results of wrong decisions made by process modelers and experts. In those cases, overlapping rules should be prevented and, when present, be repaired to ensure them to be mutual exclusive. [8] proposes a technique to detect overlapping rules and fix them to ensure their mutual exclusiveness.

4 Decision-aware Control-flow Discovery

[11] distinguishes two categories of decision-mining techniques. The approaches that falls into the first category are named *decision-annotated process mining*: first a suitable control-flow model is discovered, which is later annotated with decision rules. Every approach discussed so far is within this category. A second category aims at *decision-aware control-flow discovery*: the control-flow and decisions are discovered together in an holistic manner.

Any approach in the first category has the disadvantage that it "*imposes the initial structure on top of which the decision perspective is placed upon* [11]. This might cause important decisions to remain unrevealed: the control-flow structure is

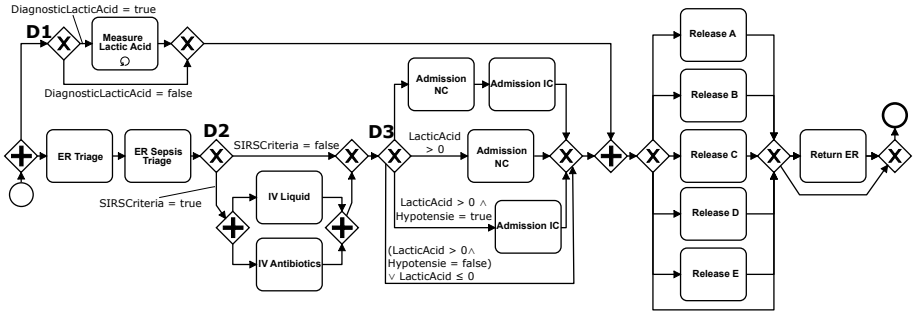

Figure 9: Overlapping, and non-overlapping decision rules discovered for the trajectory of patients in a Dutch hospital. This model is a excerpt from the model in [17, 15].

discovered without considering the overall logic of the decisions and important decision points are not made explicit.

Whereas [13] and [23] report on works to simultaneously discover data- and control-flow for declarative models (cf. the chapter of Declarative Modelling), [14] reports the only research work about decision-aware control-flow discovery of procedural models (i.e. BMPN-like) where the decision perspective is used to reveal paths in the control-flow that can be characterized by deterministic rules over the recorded attributes. If such behavior is infrequently observed it is often disregarded by control-flow discovery techniques as noise (cf. the chapter on automated process discovery). The idea is that some paths may be executed infrequently because the corresponding conditions are rarely fulfilled. These paths and the conditions are likely to be of great interest to process analysts (e.g., in the context of risks and fraud). In [14], classification techniques are employed to distinguish between such behavior and random noise.

5 Example cases and tool support

Decision mining has been applied in several domains as testified by may applications reported in literature [21, 9, 17, 14, 10, 6, 15] and several implemented tools [25, 16, 12, 5]. Here, two example cases are briefly presented. Both cases are taken from a project that was conducted in a regional hospital in The Netherlands [15].

In the first case, the trajectories of patients that are admitted to the *emergency ward* of the hospital with a suspicion for sepsis [19] were analyzed. A normative model of the expected patient trajectory control-flow was designed and, then, the overlapping decision rules method [17], yielding the model in Figure 9.

In the second case, the hospital's billing process for medical services was analyzed. Several medical services (e.g., diagnostics, treatments, etc.) are collected in a billing package, validated and invoiced. However, in some cases the billing package is *reopened*, *rejected*, or *canceled* and further changes are needed. Figure 10 shows a process model discovered by the decision-aware control-flow discovery method. Six infrequent control-flow paths that are associated with specific data attribute values have

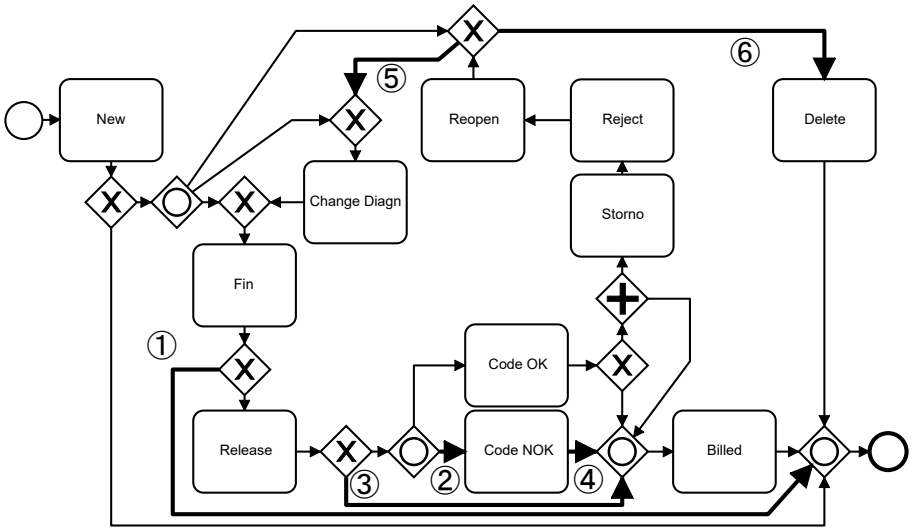

Figure 10: Process model discovered by the decision-aware process discovery method [14] for a hospital billing process [14].

been included in the process model. Some of these paths should not be disregarded as noise since they are related to cases with rework. For example, the path between RELEASE and CODE NOK (②) occurs mostly for two specific *caseType* values. According to a domain expert both case types correspond to exceptional cases: one is used for intensive care and the other for cases in which the code cannot be obtained (Code NOK). Another example, is the path from Code NOK to Billed (④), which is also related to the *caseType* attribute as well as to the medical *specialty* attribute. This path is interesting since it could not be explained by the domain expert.

6 Conclusion

This chapter has discussed the importance of the *decision perspective* and illustrates several techniques that aims to explain how the choices are driven by additional data associated with the process. These techniques leverage on the additional information recorded in data attributes (also denoted as event payload) of the event log to discover process models and enhance existing ones so that the models explain the decisions that drive the process executions. It was illustrated that these choices can be represented as annotations of process models or, alternatively, the decision can be modelled through separate DMN models that highlight the decision-related aspects.

As discussed, the application to real-life cases has illustrated that the maturity of the decision-mining approach has reached a level that certainly allows its application to production environments. While it is certainly necessary to extend the class of decisions that can be mined, the major drawback is that the decisions are mining locally

without considering the other decisions discovered at different points. This means that decision rules at different decision points can be conflicting, leading to models that, if actually enacted, would cause a certain number of executions to be unable to properly terminate. As future work, it is crucial to check whether the models enriched with decision rules are correct and to ensure to some extend that decisions are not conflicting, thus leading to deadlocks.

References

[1] Decision model and notation (DMN) v1.1, 2016.

[2] A Adriansyah. *Aligning Observed and Modeled Behavior*. PhD thesis, Eindhoven University of Technology, 2014.

[3] Kimon Batoulis, Andreas Meyer, Ekaterina Bazhenova, Gero Decker, and Mathias Weske. Extracting decision logic from process models. In *CAiSE 2015*, volume 9097 of *LNCS*, pages 349–366. Springer, 2015.

[4] Ekaterina Bazhenova, Susanne Bülow, and Mathias Weske. Discovering decision models from event logs. In *BIS 2016*, volume 255 of *LNBIP*, pages 237–251. Springer, 2016.

[5] Ekaterina Bazhenova, Stephan Haarmann, Sven Ihde, Andreas Solti, and Mathias Weske. Discovery of fuzzy DMN decision models from event logs. In *CAiSE 2017*, volume 10253 of *LNCS*, pages 629–647. Springer, 2017.

[6] Ekaterina Bazhenova and Mathias Weske. Deriving decision models from process models by enhanced decision mining. In *BPM 2015 Workshops*, pages 444–457. Springer, 2016.

[7] J. C. Bose, R. S. Mans, and Wil M. P. van der Aalst. Wanna improve process mining results? In *CIDM 2013*, pages 127–134. IEEE, 2013.

[8] Diego Calvanese, Marlon Dumas, lari Laurson, Fabrizio M. Maggi, Marco Montali, and Irene Teinemaa. Semantics and analysis of DMN decision tables. In *BPM 2016*, volume 9850 of *LNCS*, pages 217–233. Springer, 2016.

[9] Massimiliano de Leoni and Wil M. P. van der Aalst. Data-aware process mining: Discovering decisions in processes using alignments. In *SAC 2013*, pages 1454–1461. ACM, 2013.

[10] Johannes De Smedt, Faruk Hasic, Seppe vanden Broucke, and Jan Vanthienen. Towards a holistic discovery of decisions in process-aware information systems. In *BPM 2017*, volume 10445 of *LNCS*. Springer, 2017.

[11] Johannes De Smedt, Seppe K. L. M. vanden Broucke, Josue Obregon, Aekyung Kim, Jae-Yoon Jung, and Jan Vanthienen. Decision mining in a broader context: An overview of the current landscape and future directions. In *BPM 2016 Workshops*, volume 281 of *LNBIP*, pages 197–207. Springer, 2017.

[12] Anna A. Kalenkova, Massimiliano de Leoni, and Wil M. P. van der Aalst. Discovering, analyzing and enhancing BPMN models using prom. In *BPM 2014 Demos*, volume 1295 of *CEUR Workshop Proceedings*, page 36. CEUR-WS.org, 2014.

[13] Fabrizio Maria Maggi, Marlon Dumas, Luciano García-Bañuelos, and Marco Montali. Discovering data-aware declarative process models from event logs. In *BPM 2013*, volume 8094 of *LNCS*, pages 81–96. Springer, 2013.

[14] F. Mannhardt, M. de Leoni, H. A. Reijers, and W. M. P. van der Aalst. Data-driven process discovery - revealing conditional infrequent behavior from event logs. In *CAiSE 2017*, volume 10253 of *LNCS*, pages 545–560, 2017.

[15] Felix Mannhardt. *Multi-perspective Process Mining*. PhD thesis, Eindhoven University of Technology, 2018.

[16] Felix Mannhardt, Massimiliano de Leoni, and Hajo A. Reijers. The multi-perspective process explorer. In *BPM 2015 Demos*, volume 1418 of *CEUR Workshop Proceedings*, pages 130–134. CEUR-WS.org, 2015.

[17] Felix Mannhardt, Massimiliano de Leoni, Hajo A. Reijers, and Wil M. P. van der Aalst. Decision mining revisited - discovering overlapping rules. In *CAiSE 2016*, volume 9694 of *LNCS*, pages 377–392. Springer, 2016.

[18] Jorge Munoz-Gama. *Conformance Checking*. Springer International Publishing, 2018.

[19] Andrew Rhodes et al. Surviving sepsis campaign: International guidelines for management of sepsis and septic shock: 2016. *Intensive Care Med*, 43(3):304–377, 2017.

[20] Daniela Rosca and Chris Wild. Towards a flexible deployment of business rules. *Expert Syst Appl*, 23(4):385–394, 2002.

[21] A. Rozinat. *Process Mining: Conformance and Extension*. PhD thesis, Eindhoven University of Technology, Eindhoven, 2010.

[22] A. Rozinat and W. M. P. van der Aalst. Decision mining in ProM. In *BPM 2006*, volume 4102 of *LNCS*, pages 420–425. Springer, 2006.

[23] Stefan Schönig, Claudio Di Ciccio, Fabrizio Maria Maggi, and Jan Mendling. Discovery of multi-perspective declarative process models. In *ICSOC 2016*, volume 9936 of *LNCS*, pages 87–103. Springer, 2016.

[24] Wil M. P. van der Aalst. *Process Mining - Data Science in Action, Second Edition*. Springer, 2016.

[25] Boudewijn F. van Dongen, Ana Karla A. de Medeiros, H. M. W. Verbeek, A. J. M. M. Weijters, and Wil M. P. van der Aalst. The prom framework: A new era in process mining tool support. In *ICATPN 2005*, volume 3536 of *LNCS*, pages 444–454. Springer, 2005.