Discovering Process Models from Uncertain Event Data*

Marco Pegoraro[0000−0002−8997−7517], Merih Seran Uysal[0000−0003−1115−6601], and Wil M.P. van der Aalst[0000−0002−0955−6940]

Process and Data Science Group (PADS)
Department of Computer Science, RWTH Aachen University, Aachen, Germany
{pegoraro,uysal,wvdaalst}@pads.rwth-aachen.de
http://www.pads.rwth-aachen.de/

Abstract. Modern information systems are able to collect event data in the form of *event logs*. Process mining techniques allow to discover a model from event data, to check the conformance of an event log against a reference model, and to perform further process-centric analyses. In this paper, we consider uncertain event logs, where data is recorded together with explicit uncertainty information. We describe a technique to discover a directly-follows graph from such event data which retains information about the uncertainty in the process. We then present experimental results of performing inductive mining over the directly-follows graph to obtain models representing the certain and uncertain part of the process.

Keywords: Process Mining · Process Discovery · Uncertain Data.

1 Introduction

With the advent of digitalization of business processes and related management tools, *Process-Aware Information Systems* (PAISs), ranging from ERP/CRM-systems to BPM/WFM-systems, are widely used to support operational administration of processes. The databases of PAISs containing event data can be queried to obtain *event logs*, collections of recordings of the execution of activities belonging to the process. The discipline of *process mining* aims to synthesize knowledge about *processes* via the extraction and analysis of execution logs.

When applying process mining in real-life settings, the need to address anomalies in data recording when performing analyses is omnipresent. A number of such anomalies can be modeled by using the notion of uncertainty: *uncertain event logs* contain, alongside the event data, some attributes that describe a certain level of uncertainty affecting the data. A typical example is the timestamp information: in many processes, specifically the ones where data is in part manually recorded, the timestamp of events is recorded with low precision (e.g., specifying

* In *International Workshop on Business Process Intelligence* (BPI 2019). Please do not print this document unless strictly necessary.

only the day of occurrence). If multiple events belonging to the same case are recorded within the same time unit, the information regarding the event order is lost. This can be modeled as uncertainty of the timestamp attribute by assigning a time interval to the events. Another example of uncertainty are situations where the activity label is unrecorded or lost, but the events are associated with specific resources that carried out the corresponding activity. In many organizations, each resource is authorized to perform a limited set of activities, depending on her role. In this case, it is possible to model the absence of activity labels associating every event with the set of possible activities which the resource is authorized to perform.

Usually, information about uncertainty is not natively contained into a log: event data is extracted from information systems as activity label, timestamp and case id (and possibly additional attributes), without any sort of meta-information regarding uncertainty. In some cases, a description of the uncertainty in the process can be obtained from background knowledge. Information translatable to uncertainty such as the one given above as example can, for instance, be acquired from an interview with the process owner, and then inserted in the event log with a pre-processing step. Research efforts regarding how to discover uncertainty in a representation of domain knowledge and how to translate it to obtain an uncertain event log are currently ongoing.

Uncertainty can be addressed by filtering out the affected events when it appears sporadically throughout an event log. Conversely, in situations where uncertainty affects a significative fraction of an event log, filtering away uncertain event can lead to information loss such that analysis becomes very difficult. In this circumstance, it is important to deploy process mining techniques that allow to mine information also from the uncertain part of the process.

In this paper, we aim to develop a process discovery approach for uncertain event data. We present a methodology to obtain *Uncertain Directly-Follows Graphs* (UDFGs), models based on directed graphs that synthesize information about the uncertainty contained in the process. We then show how to convert UDFGs in models with execution semantics via filtering on uncertainty information and inductive mining.

The remainder of the paper is structured as follows: in Section 2 we present relevant previous work. In Section 3, we provide the preliminary information necessary for formulating uncertainty. In Section 4, we define the uncertain version of directly-follows graphs. In Section 5, we describe some examples of exploiting UDFGs to obtain executable models. Section 6 presents some experiments. Section 7 proposes future work and concludes the paper.

2 Related Work

In a previous work [9], we proposed a taxonomy of possible types of uncertainty in event data. To the best of our knowledge, no previous work addressing explicit uncertainty currently exist in process mining. Since usual event logs do not contain any hint regarding misrecordings of data or other anomalies, the notion of "noise" or "anomaly" normally considered in process discovery refers to outlier behavior. This is often obtained by setting thresholds to filter out the behavior not considered for representation in the resulting process model. A variant of the Inductive Miner by Leemans et al. [6] considers only directly-follows relationships appearing with a certain frequency. In general, a direct way to address infrequent behavior on the event level is to apply on it the concepts of support and confidence, widely used in association rule learning [5]. More sophisticated techniques employ infrequent pattern detection employing a mapping between events [8] or a finite state automaton [4] mined from the most frequent behavior.

Although various interpretations of uncertain information can exist, this paper presents a novel approach that aims to represent uncertainty explicitly, rather than filtering it out. For this reason, existing approaches to identify noise cannot be applied to the problem at hand.

3 Preliminaries

To define uncertain event data, we introduce some basic notations and concepts, partially from [2]:

Definition 1 (Power Set). *The power set of a set A is the set of all possible subsets of A, and is denoted with $\mathcal{P}(A)$. $\mathcal{P}_{NE}(A)$ denotes the set of all the non-empty subsets of A: $\mathcal{P}_{NE}(A) = \mathcal{P}(A) \setminus \{\emptyset\}$.*

Definition 2 (Sequence). *Given a set X, a finite* sequence *over X of length n is a function $s \in X^* : \{1, \dots, n\} \to X$, typically written as $s = \langle s_1, s_2, \dots, s_n \rangle$. For any sequence s we define $|s| = n$, $s[i] = s_i$, $\mathcal{S}_s = \{s_1, s_2, \dots, s_n\}$ and $x \in s \iff x \in \mathcal{S}_s$. Over the sequences s and s' we define $s \cup s' = \{a \in s\} \cup \{a \in s'\}$.*

Definition 3 (Directed Graph). *A* directed graph *$G = (V, E)$ is a set of* vertices *V and a set of directed* edges *$E \subseteq V \times V$. We denote with $\mathcal{U}_G$ the universe of such directed graphs.*

Definition 4 (Bridge). *An edge $e \in E$ is called a* bridge *if and only if the graph becomes disconnected if e is removed: there exists a partition of V into V' and V'' such that $E \cap ((V' \times V'') \cup (V'' \times V')) = \{e\}$. We denote with $E_B \subseteq E$ the set of all such bridges over the graph $G = (V, E)$.*

Definition 5 (Path). *A* path *over a graph $G = (V, E)$ is a sequence of vertices $p = \langle v_1, v_2, \dots v_n \rangle$ with $v_1, \dots, v_n \in V$ and $\forall_{1 \leq i \leq n-1} (v_i, v_{i+1}) \in E$. $P_G(v, w)$ denotes the set of all paths connecting v and w in G. A vertex $w \in V$ is* reachable *from $v \in V$ if there is at least one path connecting them: $|P_G(v, w)| > 0$.*

Definition 6 (Transitive Reduction). *A* transitive reduction *of a graph $G = (V, E)$ is a graph $\rho(G) = (V, E')$ with the same reachability between vertices and a minimal number of edges. $E' \subseteq E$ is a smallest set of edges such that $|P_{\rho(G)}(v, w)| > 0 \implies |P_G(v, w)| > 0$ for any $v, w \in V$.*

In this paper, we consider *uncertain event logs*. These event logs contain uncertainty information explicitly associated with event data. A taxonomy of different kinds of uncertainty and uncertain event logs has been presented in [9] which it distinguishes between two main classes of uncertainty. *Weak uncertainty* provides a probability distribution over a set of possible values, while *strong uncertainty* only provides the possible values for the corresponding attribute.

We will use the notion of *simple uncertainty*, which includes strong uncertainty on the control-flow perspective: activities, timestamps, and indeterminate events. An example of a simple uncertain trace is shown in Table 1. Event e_1 has been recorded with two possible activity labels (a or c), an example of strong uncertainty on activities. Some events, e.g. e_2, do not have a precise timestamp but a time interval in which the event could have happened has been recorded: in some cases, this causes the loss of the precise order of events (e.g. e_1 and e_2). These are examples of strong uncertainty on timestamps. As shown by the "?" symbol, e_3 is an indeterminate event: it has been recorded, but it is not guaranteed to have happened.

Table 1. An example of simple uncertain trace.

Case ID	Event ID	Activity	Timestamp	Event Type
0	e_1	{a, c}	[2011-12-02T00:00 2011-12-05T00:00]	!
0	e_2	{a, d}	[2011-12-03T00:00 2011-12-05T00:00]	!
0	e_3	{a, b}	2011-12-07T00:00	?
0	e_4	{a, b}	[2011-12-09T00:00 2011-12-15T00:00]	!
0	e_5	{b, c}	[2011-12-11T00:00 2011-12-17T00:00]	!
0	e_6	{b}	2011-12-20T00:00	!

Definition 7 (Universes). *Let* $\mathcal{U}_E$ *be the set of all the* event identifiers. *Let* $\mathcal{U}_C$ *be the set of all* case ID identifiers. *Let* $\mathcal{U}_A$ *be the set of all the* activity identifiers. *Let* $\mathcal{U}_T$ *be the totally ordered set of all the* timestamp identifiers. *Let* $\mathcal{U}_O = \{!, ?\}$, *where the "!" symbol denotes* determinate events, *and the "?" symbol denotes* indeterminate events.

Definition 8 (Simple uncertain traces and logs). $\sigma \in \mathcal{P}_{NE}(\mathcal{U}_E \times \mathcal{P}_{NE}(\mathcal{U}_A) \times \mathcal{U}_T \times \mathcal{U}_T \times \mathcal{U}_O)$ *is a* simple uncertain trace *if for any* $(e_i, A, t_{min}, t_{max}, u) \in \sigma$, $t_{min} < t_{max}$ *and all the event identifiers are unique.* $\mathcal{T}_U$ *denotes the universe of simple uncertain traces.* $L \in \mathcal{P}(\mathcal{T}_U)$ *is a* simple uncertain log *if all the event identifiers in the log are unique. Over the uncertain event* $e = (e_i, A, t_{min}, t_{max}, o) \in \sigma$ *we define the following projection functions:* $\pi_A(e) = A$, $\pi_{t_{min}}(e) = t_{min}$, $\pi_{t_{max}}(e) = t_{max}$ *and* $\pi_o(e) = o$. *Over* $L \in \mathcal{P}(\mathcal{T}_U)$ *we define the following projection function:* $\Pi_A(L) = \bigcup_{\sigma \in L} \bigcup_{e \in \sigma} \pi_A(e)$.

The behavior graph is a structure that summarizes information regarding the uncertainty contained in a trace. Namely, two vertices are linked by an edge if their corresponding events may have happened one immediately after the other.

Definition 9 (Behavior Graph). *Let $\sigma \in \mathcal{T}_U$ be a simple uncertain trace. A* behavior graph $\beta \colon \mathcal{T}_U \to \mathcal{U}_G$ *is the transitive reduction of a directed graph $\rho(G)$, where $G = (V, E) \in \mathcal{U}_G$ is defined as:*

- $V = \{e \in \sigma\}$
- $E = \{(v, w) \mid v, w \in V \wedge \pi_{t_{max}}(v) < \pi_{t_{min}}(w)\}$

Notice that the behavior graph is obtained from the transitive reduction of an acyclic graph, and thus is unique. The behavior graph for the trace in Table 1 is shown in Figure 1.

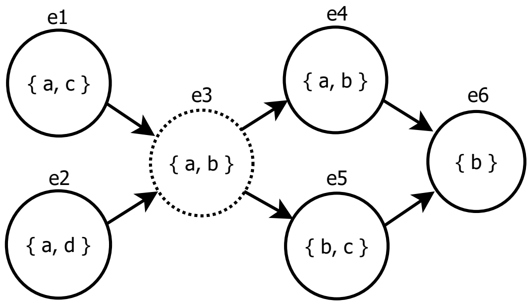

Fig. 1. The behavior graph of the uncertain trace given in Table 1. Each vertex represents an uncertain event and is labeled with the possible activity label of the event. The dotted circle represents an indeterminate event (may or may not have happened).

4 Uncertain DFGs

The definitions shown in Section 3 allow us to introduce some fundamental concepts necessary to perform discovery in an uncertain setting. Let us define a measure for the frequencies of single activities. In an event log without uncertainty the frequency of an activity is the number of events that have the corresponding activity label. In the uncertain case, there are events that can have multiple possible activity labels. For a certain activity $a \in \mathcal{U}_A$, the minimum activity frequency of a is the number of events that certainly have A as activity label and certainly happened; the maximum activity frequency is the number of events that may have A as activity label.

Definition 10 (Minimum and maximum activity frequency). *The* minimum *and* maximum activity frequency $\#_{min} \colon \mathcal{T}_U \times \mathcal{U}_A \rightarrow \mathbb{N}$ *and* $\#_{max} \colon \mathcal{T}_U \times \mathcal{U}_A \rightarrow \mathbb{N}$ *of an activity* $a \in \mathcal{U}_A$ *in regard of an uncertain trace* $\sigma \in \mathcal{T}_U$ *are defined as:*

- $\#_{min}(\sigma, a) = |\{e \in \sigma \mid \pi_A(e) = \{a\} \wedge \pi_o(v) = !\}|$
- $\#_{max}(\sigma, a) = |\{e \in \sigma \mid a \in \pi_A(e)\}|$.

Many discovery algorithms exploit the concept of *directly-follows relationship* [1,6]. In this paper, we extend this notion to uncertain traces and uncertain event logs. An uncertain trace embeds some behavior which depends on the instantiation of the stochastic variables contained in the event attributes. Some directly-follows relationships exist in part, but not all, the possible behavior of an uncertain trace. As an example, consider events e_3 and e_5 in the uncertain trace shown in Table 1: the relationship "a is directly followed by b" appears once only if e_3 actually happened immediately before e_5 (i.e., e_4 did not happen in-between), and if the activity label of e_3 is a b (as opposed to c, the other possible label). In all the behavior that does not satisfy these conditions, the directly-follows relation does not appear on e_3 and e_5.

Let us define as *realizations* all the possible certain traces that are obtainable by choosing a value among all possible ones for an uncertain attribute of the uncertain trace. For example, some possible realizations of the trace in Table 1 are $\langle a, d, b, a, c, b\rangle$, $\langle a, a, a, a, b, b\rangle$, and $\langle c, a, c, b, b\rangle$. We can express the strength of the directly-follows relationship between two activities in an uncertain trace by counting the minimum and maximum number of times the relationship can appear in one of the possible realizations of that trace. To this goal, we exploit some structural properties of the behavior graph in order to obtain the minimum and maximum frequency of directly-follows relationships in a simpler manner.

A useful property to compute the minimum number of occurrences between two activities exploits the fact that parallel behavior is represented by the branching of arcs in the graph. Two connected determinate events have happened one immediately after the other if the graph does not have any other parallel path: if two determinate events are connected by a bridge, they will certainly happen in succession. This property is used to define a *strong sequential relationship*.

The next property accounts for the fact that, by construction, uncertain events corresponding to nodes in the graph not connected by a path can happen in any order. This follows directly from the definition of the edges in the graph, together with the transitivity of $\mathcal{U}_T$ (which is a totally ordered set). This means that two disconnected nodes v and w may account for one occurrence of the relation "$\pi_A(v)$ is directly followed by $\pi_A(w)$". Conversely, if w is reachable from v, the directly-follows relationship may be observed if all the events separating v from w are indeterminate (i.e., there is a chance that no event will interpose between the ones in v and w). This happens for vertices e_2 and e_4 in the graph in Figure 1, which are connected by a path and separated only by vertex e_3, which is indeterminate. This property is useful to compute the maximum number of directly-follows relationships between two activities, leading to the notion of *weak sequential relationship.*

Definition 11 (Strong sequential relationship). *Given a behavior graph* $\beta = (V, E)$ *and two vertices* $v, w \in V$*,* v *is in a* strong sequential relationship *with* w *(denoted by* $v \blacktriangleright_\beta w$*) if and only if* $\pi_o(v) = !$ *and* $\pi_o(w) = !$ *(*v *and* w *are both determinate) and there is a bridge between them:* $(v, w) \in E_B$*.*

Definition 12 (Weak sequential relationship). *Given a behavior graph* $\beta = (V, E)$ *and two vertices* $v, w \in V$*,* v *is on a* weak sequential relationship *with* w *(denoted by* $v \triangleright_\beta w$*) if and only if* $|P_\beta(w, v)| = 0$ *(*v *is unreachable from* w*) and no node in any possible path between* v *and* w*, excluding* v *and* w*, is determinate:* $\bigcup_{p \in P_\beta(v,w)} \{e \in p \mid \pi_o(e) = !\} \setminus \{v, w\} = \emptyset$*.*

Notice that if v and w are mutually unreachable they are also in a mutual weak sequential relationship. Given two activity labels, these properties allow us to extract sets of candidate pairs of vertices of the behavior graph.

Definition 13 (Candidates for minimum and maximum directly-follows frequencies). *Given two activities* $a, b \in \mathcal{U}_A$ *and an uncertain trace* $\sigma \in \mathcal{T}_U$ *and the corresponding behavior graph* $\beta(\sigma) = (V, E)$*, the* candidates for minimum and maximum directly-follows frequency $cand_{min} \colon \mathcal{T}_U \times \mathcal{U}_A \times \mathcal{U}_A \to \mathcal{P}(V \times V)$ *and* $cand_{max} \colon \mathcal{T}_U \times \mathcal{U}_A \times \mathcal{U}_A \to \mathcal{P}(V \times V)$ *are defined as:*

- $cand_{min}(\sigma, a, b) = \{(v, w) \in V \times V \mid v \neq w \wedge \pi_A(v) = \{a\} \wedge \pi_A(w) = \{b\} \wedge v \blacktriangleright_\beta w\}$
- $cand_{max}(\sigma, a, b) = \{(v, w) \in V \times V \mid v \neq w \wedge a \in \pi_A(v) \wedge b \in \pi_A(w) \wedge v \triangleright_\beta w\}$

After obtaining the sets of candidates, it is necessary to select a subset of pair of vertices such that there are no repetitions. In a realization of an uncertain trace, an event e can only have one successor: if multiple vertices of the behavior graph correspond to events that can succeed e, only one can be selected.

Consider the behavior graph in Figure 1. If we search candidates for "a is directly followed by b", we find $\mathrm{cand}_{\mathrm{min}}(\sigma, a, b) = \{(e_1, e_3), (e_2, e_3), (e_1, e_5), (e_2, e_4), (e_3, e_4), (e_3, e_5), (e_4, e_6)\}$. However, there are no realizations of the trace represented by the behavior graph that contains all the candidates; this is because some vertices appear in multiple candidates. A possible realization with the highest frequency of $a \to b$ is $\langle d, a, b, c, a, b \rangle$. Conversely, consider "$a$ is directly followed by a". When the same activity appears in both sides of the relationship, an event can be part of two different occurrences, as first member and second member; e. g., in the trace $\langle a, a, a \rangle$, the relationship $a \to a$ occurs two times, and the second event is part of both occurrences. In the behavior graph of Figure 1, the relation $a \to b$ cannot be supported by candidates (e_1, e_3) and (e_3, e_4) at the same time, because e_3 has either label a or b in a realization. But (e_1, e_3) and (e_3, e_4) can both support the relationship $a \to a$, in realizations where e_1, e_3 and e_4 all have label a.

When counting the frequencies of directly follows relationships between the activities a and b, every node of the behavior graph can appear at most once if $a \neq b$. If $a = b$, every node can appear once on each side of the relationship.

Definition 14 (Minimum directly-follows frequency). *Given $a, b \in \mathcal{U}_A$ and $\sigma \in \mathcal{T}_U$, let $R_{min} \subseteq cand_{min}(\sigma, a, b)$ be a largest set such that for any $(v, w), (v', w') \in R_{min}$, it holds:*

$$(v, w) \neq (v', w') \implies \{v, w\} \cap \{v', w'\} = \emptyset, \qquad \text{if } a \neq b$$

$$(v, w) \neq (v', w') \implies v \neq v' \wedge w \neq w', \qquad \text{if } a = b$$

The minimum directly-follows frequency $\rightsquigarrow_{min} : \mathcal{T}_U \times {\mathcal{U}_A}^2 \to \mathbb{N}$ *of two activities $a, b \in \mathcal{U}_A$ in regard of an uncertain trace $\sigma \in \mathcal{T}_U$ is defined as $\rightsquigarrow_{min} (\sigma, a, b) = |R_{min}|$.*

Definition 15 (Maximum directly-follows frequency). *Given $a, b \in \mathcal{U}_A$ and $\sigma \in \mathcal{T}_U$, let $R_{max} \subseteq cand_{max}(\sigma, a, b)$ be a largest set such that for any $(v, w), (v', w') \in R_{max}$, it holds:*

$$(v, w) \neq (v', w') \implies \{v, w\} \cap \{v', w'\} = \emptyset, \qquad \text{if } a \neq b$$

$$(v, w) \neq (v', w') \implies v \neq v' \wedge w \neq w', \qquad \text{if } a = b$$

The maximum directly-follows frequency $\rightsquigarrow_{max} : \mathcal{T}_U \times {\mathcal{U}_A}^2 \to \mathbb{N}$ *of two activities $a, b \in \mathcal{U}_A$ in regard of an uncertain trace $\sigma \in \mathcal{T}_U$ is defined as $\rightsquigarrow_{max} (\sigma, a, b) = |R_{max}|$.*

For the uncertain trace in Table 1, $\rightsquigarrow_{\text{min}} (\sigma, a, b) = 0$, because $R_{\text{min}} = \emptyset$; conversely, $\rightsquigarrow_{\text{max}} (\sigma, a, b) = 2$, because a maximal set of candidates is $R_{\text{max}} = \{(e_1, e_3), (e_4, e_6)\}$. Notice that maximal candidate sets are not necessarily unique: $R_{\text{max}} = \{(e_2, e_3), (e_4, e_6)\}$ is also a valid one.

The operator $\rightsquigarrow$ synthesizes information regarding the strength of the directly-follows relation between two activities in an event log where some events are uncertain. The relative difference between the *min* and *max* counts is a measure of how certain the relationship is when it appears in the event log. Notice that, in the case where no uncertainty is contained in the event log, *min* and *max* will coincide, and will both contain a directly-follows count for two activities.

An *Uncertain DFG* (UDFG) is a graph representation of the activity frequencies and the directly-follows frequencies; using the measures we defined, we exclude the activities and the directly-follows relations that never happened.

Definition 16 (Uncertain Directly-Follows Graph (UDFG)). *Given an event log $L \in \mathcal{P}(\mathcal{T}_U)$, the Uncertain Directly-Follows Graph $DFG_U(L)$ is a directed graph $G = (V, E)$ where:*

- $V = \{a \in \Pi_A(L) \mid \sum_{\sigma \in L} \#_{max}(\sigma, a) > 0\}$
- $E = \{(a, b) \in V \times V \mid \sum_{\sigma \in L} \rightsquigarrow_{max} (\sigma, a, b) > 0\}$

The UDFG is a low-abstraction model that, together with the data decorating vertices and arcs, gives indications on the overall uncertainty affecting activities and directly-follows relationships. Moreover, the UDFG does not filter out uncertainty: the information about the uncertain portion of a process is summarized by the data labeling vertices and edges. In addition to the elimination of the anomalies in an event log in order to identify the happy path of a process, this allows the process miner to isolate the uncertain part of a process, in order to study its features and analyze its causes. In essence however, this model has the same weak points as the classic DFG: it does not support concurrency, and if many activities happen in different order the DFG creates numerous loops that cause underfitting.

5 Inductive Mining Using Directly-Follows Frequencies

A popular process mining algorithm for discovering executable models from DFGs is the Inductive Miner [6]. A variant presented by Leemans et al. [7], the *Inductive Miner–directly-follows* (IM$_D$), has the peculiar feature of preprocessing an event log to obtain a DFG, and then discover a process tree exclusively from the graph, which can then be converted to a Petri net. This implies a high scalability of the algorithm, which has a linear computational cost over the number of events in the log, but it also makes it suited to the case at hand in this paper. To allow for inductive mining, and subsequent representation of the process as a Petri net, we introduce a form of filtering called UDFG slicing, based on four filtering parameters: act_{min}, act_{max}, rel_{min} and rel_{max}. The parameters act_{min} and act_{max} allow to filter on nodes of the UDFG, based on how certain the corresponding activity is in the log. Conversely, rel_{min} and rel_{max} allow to filter on edges of the UDFG, based on how certain the corresponding directly-follows relationship is in the log.

Definition 17 (Uncertain DFG slice). *Given an uncertain event log $L \in \mathcal{P}(\mathcal{T}_U)$, its uncertain directly-follows graph $DFG_U(L) = (V', E')$, and act_{min}, $act_{max}, rel_{min}, rel_{max} \in [0, 1]$, an* uncertain directly-follows slice *is a function $\overline{DFG}_U \colon L \to \mathcal{U}_G$ where $\overline{DFG}_U(L, act_{min}, act_{max}, rel_{min}, rel_{max}) = (V, E)$ with:*

- $V = \{a \in V' \mid act_{min} \leq \frac{\sum_{\sigma \in L} \#_{min}(\sigma, a)}{\sum_{\sigma \in L} \#_{max}(\sigma, a)} \leq act_{max}\}$
- $E = \{(a, b) \in E' \mid rel_{min} \leq \frac{\sum_{\sigma \in L} \rightsquigarrow_{min}(\sigma, a, b)}{\sum_{\sigma \in L} \rightsquigarrow_{max}(\sigma, a, b)} \leq rel_{max}\}$

A UDFG slice is an unweighted directed graph which represents a filtering performed over vertices and edges of the UDFG. This graph can then be processed by the IM$_D$.

Definition 18 (Uncertain Inductive Miner–directly-follows (UIM$_D$)). *Given an uncertain event log $L \in \mathcal{P}(\mathcal{T}_U)$ and $act_{min}, act_{max}, rel_{min}, rel_{max} \in [0, 1]$, the* Uncertain Inductive Miner–directly-follows *(UIM$_D$) returns the process tree obtained by IM$_D$ over an uncertain DFG slice: IM$_D$($\overline{DFG}_U(L, act_{min}$, $act_{max}, rel_{min}, rel_{max})$).*

The filtering parameters act_{min}, act_{max}, rel_{min}, rel_{max} allow to isolate the desired type of behavior of the process. In fact, $act_{min} = rel_{min} = 0$ and $act_{max} = rel_{max} = 1$ retain all possible behavior of the process, which is then represented in the model: both the behavior deriving from the process itself and the behavior deriving from the uncertain traces. Higher values of act_{min} and rel_{min} allow to filter out uncertain behavior, and to retain only the parts of the process observed in certain events. Vice versa, lowering act_{min} and rel_{min} allows to observe only the uncertain part of an event log.

6 Experiments

The approach described here has been implemented using the Python process mining framework PM4Py [3]. The models obtained through the Uncertain Inductive Miner–directly-follows cannot be evaluated with commonly used metrics in process mining, since metrics in use are not applicable on uncertain event data; nor other approaches for performing discovery over uncertain data exist. This preliminary evaluation of the algorithm will, therefore, not be based on measurements; it will show the effect of the UIM_D with different settings on an uncertain event log.

Let us introduce a simplified notation for uncertain event logs. In a trace, we represent an uncertain event with multiple possible activity labels by listing the labels between curly braces. When two events have overlapping timestamps, we represent their activity labels between square brackets, and we represent the indeterminate events by overlining them. For example, the trace $\langle \overline{a}, \{b, c\}, [d, e] \rangle$ is a trace containing 4 events, of which the first is an indeterminate event with label a, the second is an uncertain event that can have either b or c as activity label, and the last two events have a range as timestamp (and the two ranges overlap). The simplified representation of the trace in Table 1 is $\langle [\{a, c\}, \{a, d\}], \overline{\{a, b\}}, [\{a, b\}, \{b, c\}], b \rangle$. Let us observe the effect of the UIM_D on the following test log:

$\langle a, b, e, f, g, h \rangle^{80}, \langle a, [\{b, c\}, e], \overline{f}, g, h, i \rangle^{15}, \langle a, [\{b, c, d\}, e], \overline{f}, g, h, j \rangle^{5}$.

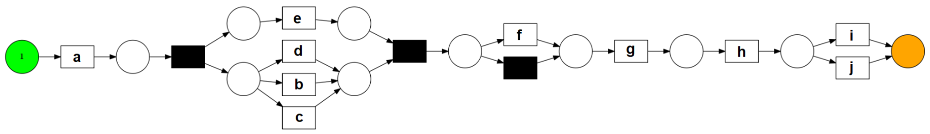

Fig. 2. UIM_D on the test log with $act_{min} = 0$, $act_{max} = 1$, $rel_{min} = 0$, $rel_{max} = 1$.

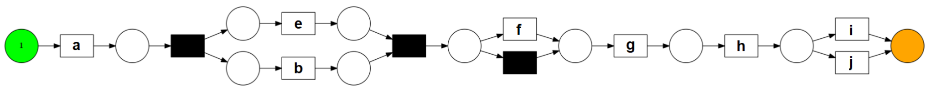

Fig. 3. UIM_D on the test log with $act_{min} = 0.6$, $act_{max} = 1$, $rel_{min} = 0$, $rel_{max} = 1$.

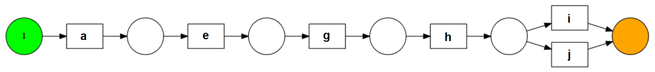

Fig. 4. UIM_D on the test log with $act_{min} = 0.9$, $act_{max} = 1$, $rel_{min} = 0$, $rel_{max} = 1$.

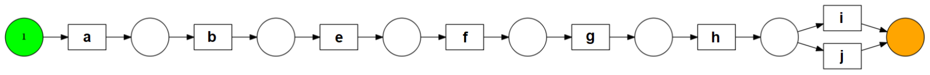

Fig. 5. UIM_D on the test log with $act_{min} = 0$, $act_{max} = 1$, $rel_{min} = 0.7$, $rel_{max} = 1$.

In Figure 2, we can see the model obtained without any filtering: it represents all the possible behavior in the uncertain log. The models in Figures 3 and 4 show the effect on filtering on the minimum number of times an activity appears in the log: in Figure 3 activities c and d are filtered out, while the model in Figure 4 only retains the activities which never appear in an uncertain event (i.e., the activities for which $\#_{min}$ is at least 90% of $\#_{max}$).

Filtering on rel_{min} has a similar effect, although it retains the most certain relationships, rather than activities, as shown in Figure 5. An even more aggressive filtering of rel_{min}, as shown in Figure 6, allows to represent only the parts of the process which are never subjected to uncertainty by being in a directly-follows relationship that has a low $\rightsquigarrow_{min}$ value.

The UIM_D allows also to do the opposite: hide certain behavior and highlight the uncertain behavior. Figure 7 shows a model that only displays the behavior which is part of uncertain attributes, while activities h, i and j – which are never part of uncertain behavior – have not been represented. Notice that g is represented even though it always appeared as a certain event; this is due to the fact that the filtering is based on relationships, and g is in a directly-follows relationship with the indeterminate event f.

7 Conclusion

In this explorative work, we present the foundations for performing process discovery over uncertain event data. We present a method that is effective in representing a process containing uncertainty by exploiting the information into an uncertain event log to synthesize an uncertain model. The UDFG is a formal description of uncertainty, rather than a method to eliminate uncertainty to observe the underlying process. This allows to study uncertainty in isolation, possibly allowing us to determine which effects it has on the process in terms

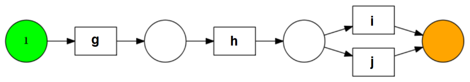

Fig. 6. UIM_D on the test log with $act_{min} = 0$, $act_{max} = 1$, $rel_{min} = 0.9$, $rel_{max} = 1$.

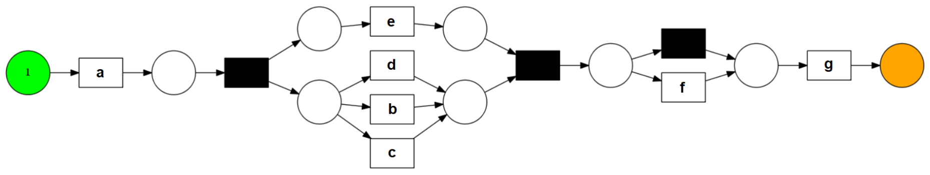

Fig. 7. UIM_D on the test log with $act_{min} = 0$, $act_{max} = 1$, $rel_{min} = 0$, $rel_{max} = 0.8$.

of behavior, as well as what are the causes of its appearance. We also present a method to filter the UDFG, obtaining a graph that represents a specific perspective of the uncertainty in the process; this can be then transformed in a model that is able to express concurrency using the UIM_D algorithm.

This approach has a number of limitations that will need to be addressed in future work. An important research direction is the formal definition of metrics and measures over uncertain event logs and process models, in order to allow for a quantitative evaluation of the quality of this discovery algorithm, as well as other process mining methods over uncertain logs. Another line of research can be the extension to the weakly uncertain event data (i.e., including probabilities) and the extension to event logs also containing uncertainty related to case IDs.

References

1. Van der Aalst, W., Weijters, T., Maruster, L.: Workflow mining: Discovering process models from event logs. IEEE Transactions on Knowledge and Data Engineering **16**(9), 1128–1142 (2004)
2. Van der Aalst, W.M.: Process mining: data science in action. Springer (2016)
3. Berti, A., van Zelst, S.J., van der Aalst, W.: Process Mining for Python (PM4Py): Bridging the Gap Between Process- and Data Science. In: International Conference on Process Mining - Demo Track. IEEE (2019)
4. Conforti, R., La Rosa, M., ter Hofstede, A.H.: Filtering out infrequent behavior from business process event logs. IEEE Transactions on Knowledge and Data Engineering **29**(2), 300–314 (2017)
5. Hornik, K., Grün, B., Hahsler, M.: arules – a computational environment for mining association rules and frequent item sets. Journal of Statistical Software **14**(15), 1–25 (2005)
6. Leemans, S.J., Fahland, D., van der Aalst, W.M.: Discovering block-structured process models from event logs-a constructive approach. In: International conference on applications and theory of Petri nets and concurrency. pp. 311–329. Springer (2013)
7. Leemans, S.J., Fahland, D., Van der Aalst, W.M.: Scalable process discovery and conformance checking. Software & Systems Modeling **17**(2), 599–631 (2018)
8. Lu, X., Fahland, D., van den Biggelaar, F.J., van der Aalst, W.M.: Detecting deviating behaviors without models. In: International Conference on Business Process Management. pp. 126–139. Springer (2016)
9. Pegoraro, M., van der Aalst, W.M.: Mining uncertain event data in process mining. In: International Conference on Process Mining. IEEE (2019)