Toward an Automated Labeling of Event Log Attributes

Abbad Andaloussi, Amine; Burattin, Andrea; Weber, Barbara

Published in:
BPMDS 2018 : Business Process Modeling, Development, and Support

Publication date:
2018

Document Version
Peer reviewed version

Link back to DTU Orbit

Citation (APA):
Abbad Andaloussi, A., Burattin, A., & Weber, B. (2018). Toward an Automated Labeling of Event Log Attributes. In BPMDS 2018 : Business Process Modeling, Development, and Support

Toward an Automated Labeling of Event Log Attributes

Amine A. Andaloussi, Andrea Burattin, Barbara Weber

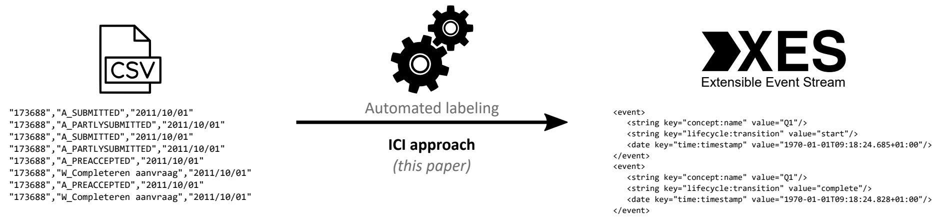
DTU Compute, Software Engineering
Technical University of Denmark
2800 Kgs. Lyngby, Denmark
{amab, andbur, bweb}@dtu.dk

Abstract. Process mining aims at exploring the data produced by executable business processes to mine the underlying control-flow and data-flow. Most of the process mining algorithms assume the existence of an event log with a certain maturity level. Unfortunately, the logs provided by process unaware information systems often do not comply with the required maturity level, since they lack the notion of process instance, also referred in process mining as "case id". Without a proper identification of the case id attribute in log files, the outcome of process mining algorithms is unpredictable. This paper proposes a new approach that aims to overcome this challenge by automatically inferring the case id attribute from log files. The approach has been implemented as a ProM plugin and evaluated with several real-world event logs. The results demonstrate a high accuracy in inferring the case id attribute.

1 Introduction

The event logs produced by information systems provide insights into the executed process instances and allow to perceive the individual behaviour of each of them. In process mining, the control-flow is extracted from the recorded behavior which represents the order in which events were executed, and the data-flow is extracted from the correlation among events' attributes. However, to explore process mining capabilities to mine the individual behaviour of each process instance from an event log, it is necessary to distinguish and isolate each recorded process instances. This requirement gets more complicated in case of concurrent execution of process instances which is one of the fundamental principles of designing modern BPM systems [12]. Under this circumstance, the correlation among events becomes uncertain, as two successive events in the log may belong to different process instances. As solution, a case identifier (*case id*) should be attributed to each single process execution. A case identifier is assigned to events with the same attribute value for all events belonging to the same process instance.

The availability of the case id attribute in an event log depends on its level of maturity. The process mining manifesto [10] introduced a maturity ranking of event logs depending on the level of logging information they provide. Among



```
"173688","A_SUBMITTED","2011/10/01"
"173688","A_PARTLYSUBMITTED","2011/10/01"
"173688","A_SUBMITTED","2011/10/01"
"173688","A_PARTLYSUBMITTED","2011/10/01"
"173688","A_PREACCEPTED","2011/10/01"
"173688","W_Completeren aanvraag","2011/10/01"
"173688","A_PREACCEPTED","2011/10/01"
"173688","W_Completeren aanvraag","2011/10/01"
```

```
<event>
   <string key="concept:name" value="Q1"/>
   <string key="lifecycle:transition" value="start"/>
   <date key="time:timestamp" value="1970-01-01T09:18:24.685+01:00"/>
</event>
<event>
   <string key="concept:name" value="Q1"/>
   <string key="lifecycle:transition" value="complete"/>
   <date key="time:timestamp" value="1970-01-01T09:18:24.828+01:00"/>
</event>
```

Fig. 1. Context of the approach presented in this paper

the maturity criteria used for the ranking is the notion of case id that should be explicitly mentioned in the log. Burattin in [4] characterized companies according to the process awareness of their working methodology and to the process awareness of the software system they use. In this context, several process aware companies use process unaware systems where the notion of case id is not explicit. The approach presented in this paper is beneficial for the following types of process unaware software systems: *(a)* Customer Relationship Management systems (CRM) where the process data is spread over a complex relational database; *(b)* Internet of Things (IoT) environments where sensor data is recorded on external log files; *(c)* Document Management Systems (DMS) where digital documents are stored and managed (if the company works in a process oriented manner, metadata of each documents are likely to contain information concerning the case id, such as client number or invoice number). The event logs produced by those systems have a low maturity ranking since they all lack the notion of a case id. Moreover, the log data is usually spread over dozens of attributes, making it very impractical to manually try them all before finding a correct match with the case id attribute. As the identification of the case id attribute is crucial for most process mining analyses, an approach to infer the case id from such software system logs is valuable.

The context of the approach is depicted in Figure 1. The purpose is to automate the labeling in the process of converting log files from CSV format to XES[1] format. This paper addresses the challenge of inferring the case id as an initial stride toward a generic approach allowing to infer all the other relevant event log attributes (i.e., activity name, resource). By exploring the control-flow discovery quality dimensions a new approach to infer the case ids from event logs (Infer Case Id, abbreviated as ICI) is introduced and evaluated using both synthetic and real-world event logs. The rest of the paper is structured as follows: Section 2 discusses the related work. Section 3 presents the event log labeling approach along with the concepts used to explain it. Section 4 describes the implementation. Section 5 evaluates the proposed approach, and Section 6 discusses the obtained results. Finally, Section 7 concludes the paper.

[1] See `https://standards.ieee.org/findstds/standard/1849-2016.html`

2 Related work

The challenge of inferring case ids from event logs has obtained little attention from the BPM community. The reason is that most of the literature introducing new process mining techniques assume the existence of labelled log files where the case id is known beforehand. This paper drops such assumption, and goes on a quest for automatically inferring case ids. By looking at the existing related work, it is notable that few publications [6, 14, 5, 2] have already raised this challenge and proposed different approaches from different perspectives to solve the problem.

Ferreira et al., in [6], proposed an approach to transform an unlabeled log into a labelled one using the Expectation-Maximization technique. This aims at finding a solution that converges to a local maximum of a likelihood function. This approach is considered by the authors as generic and executable in different environments. However, inferring case ids using this technique might be subject to uncertainty because the first-order Markovian model used is unable to represent some work-flow patterns such as loops and parallelism. An enhanced approach, suggested by Walicki and Ferreira [14], suggests a sequence partitioning technique. However, the proposed technique shares the same limitations as the previous one [6] since it is limited to only simple work-flow patterns, thus it does not support loops and parallelism.

Bayomie et al. in [2] proposed an approach to infer the case identifiers from unlabeled event logs. The approach requires the reference process model used to document the business process, which is often part of the documentation package delivered at design time. The reference process model is used to generate a causal behavioural profile [15]. The latter is used together with time heuristics inferred from the event log to build a decision tree where each node represents an event from the event log and carries its conditional probability of belonging to the same case as its parent node. Bayomie's approach aims at generating a set of labelled event logs listed according to a ranking score used to indicate their degree of trust. The approach explores the data-flow correlations (time heuristics) and control-flow correlations (causal behavioural profile) between events to group them by process instance.

Burattin and Vigo in [5] share the same assumption as the ICI approach, by considering the case id as a hidden attribute inside the log. The authors justify their assumption by the fact that it is general enough for a broad range of real-world event logs. Their proposed approach consists of filtering a set of event attributes considered as candidates for representing the case id in an event log. The filtering is done to reduce the search space by applying some selection heuristics such as selecting only the attributes with specific data types (i.e., ignoring timestamps), and using regular expression constraints as a selection criteria. Afterwards, the approach exploits the amount of data shared between attributes to construct chains, such that each chain links all similar attributes' values across the log. The case ids are then, inferred by choosing the chain with maximal length and minimal number of crossed attributes. By reducing the search space, the approach aims at finding a set of possible combinations of attribute that might represent the case id. However, it relies entirely on the similarity of attributes; thus, its accuracy is limited to a specific range of logs.

The ICI approach presented in this paper overcomes all the challenges of the previously cited approaches. Indeed, it does not require any reference process model nor similar heuristics to infer the case ids. The aim is to introduce a generic approach to infer the case id – but other attributes as well – from event logs using the control-flow discovery quality dimensions.

3 Method

The ICI approach automates the event log labeling process. Section 3.1 provides a formal definition of the notations used to describe the approach, Section 3.2 presents the control-flow quality dimensions, Section 3.3 highlights the underlying assumptions, Section 3.4 presents the approach, and Section 3.5 illustrates the ICI approach by providing a running example.

3.1 Preliminaries

In this section, formal definitions for *sequence*, *event*, *raw event*, *trace, case*, and *event log* are provided. These definitions are combined from existing work available in the literature [8].

Definition 1 (Sequence). *Given a set $\mathcal{A}$, a* finite sequence *over $\mathcal{A}$ of length n is a mapping $s \in ([1,n] \subset \mathbb{N}) \to \mathcal{A}$ and can be represented as a string, i.e., $s = \langle s_1, s_2, \ldots, s_n \rangle$, where $s_i \in \mathcal{A}$. We write $s \in \mathcal{A}^*$ to indicate that the sequence s contains elements from $\mathcal{A}$. Over a sequence s the following functions are defined:*

- *Selection operator: $s(i) = s_i$, $\forall\, 1 \leq i \leq n$;*
- *Size operator: $|s| = n$ (i.e., the length of the sequence).*

Definition 2 (Event). *Given any notion of process, let $\mathcal{A}$ be the set of all possible activities contained in the process, let $\mathcal{C}$ be the set of all possible case ids (i.e., the set of all possible identifiers of process instances), and let $\mathcal{D}_1, \ldots, \mathcal{D}_m$ be the set of additional data attributes characterizing each event executed. An* event *is a tuple $e = (a, c, t, d_1, \ldots, d_m)$, where:*

- *$a \in \mathcal{A}$ represents the activity associated to the event;*
- *$c \in \mathcal{C}$ represents the case id;*
- *$t \in \mathbb{N}$ represents the timestamp;*
- *$d_1, \ldots, d_m$ is a list of additional (and optional) event attributes, where $\forall\, 1 \leq i \leq m, d_i \in \mathcal{D}_i \cup \{\perp\}$.*

$\mathcal{E} = \mathcal{A} \times \mathcal{C} \times \mathbb{N} \times \mathcal{D}^{\perp}{}_1 \times \ldots \times \mathcal{D}^{\perp}{}_m$ is called the event universe. In an event e, the following projection functions are defined: $\pi_a(e) = a$, $\pi_c(e) = c$, $\pi_t(e) = t$, and $\pi_{d_i}(e) = d_i, \forall\, 1 \leq i \leq m$. If e does not contain the attribute value d_i for some $i \in [1,m] \subset \mathbb{N}$, $\pi_{d_i}(e) = \perp$.

The event definition (cf. Def. 2) assumes the existence of a case id $c \in \mathcal{C}$ in order to have the tuple $e = (a, c, t, d_1, \dots, d_m)$. Since the ICI approach assumes that the case id is unknown a-priori, it is necessary to define a *Raw Event* as an event with unknown case id.

Definition 3 (Raw Event). *A* Raw Event *is a tuple* $\hat{e} = (a, t, d_1, \dots, d_k)$, *where:*

- $a \in \mathcal{A}$ *represents the activity associated to the event;*
- $t \in \mathbb{N}$ *represents the timestamp;*
- $d_1, \dots, d_k$ *is a list of raw event attributes.*

$\hat{\mathcal{E}} = \mathcal{A} \times \mathbb{N} \times \mathcal{D}_1 \times \dots \times \mathcal{D}_k$ *is called the raw event universe. In a raw event* $\hat{e}$, *the following projection functions are defined:* $\pi_a(\hat{e}) = a$, $\pi_t(\hat{e}) = t$, *and* $\pi_{d_i}(\hat{e}) = d_i, \forall\, 1 \leq i \leq k$. *If* $\hat{e}$ *does not contain the attribute value* d_i *for some* $i \in [1, k] \subset \mathbb{N}$, $\pi_{d_i}(\hat{e}) = \perp$.

Definition 4 (Trace, Case). *A* trace *is defined as a finite sequence of events* $\sigma_c = \langle e_1, e_2, \dots, e_n \rangle \in \mathcal{E}^*$ *such that* $\exists c \in \mathcal{C}\ \forall\, 1 \leq i \leq |\sigma|, \pi_c(e_i) = c \wedge \forall\, 1 \leq j < |\sigma_c|$, $\pi_t(\sigma_c(e_j)) \leq \pi_t(\sigma_c(e_{j+1}))$. *In the context of this paper, each* case *is a grouping of events belonging to the same process execution and having same case id. Thus, each* case *is a distinct* trace. *Additionally, the sequence of events in a trace is ordered according to their timestamp.*

Definition 5 (Event Log). *An* Event Log L *is defined as a set of events such that* $L \subseteq \mathcal{E}$. *Please note that it is possible to group events based on their case id in order to identify traces.*

3.2 Control-flow Quality Dimensions

This section describes the control-flow quality dimensions considered by the ICI approach. The availability of an event log allows generating different process models depending on the discovery algorithm used. The generated models can be evaluated based on the following four quality dimensions: *Fitness*, *Precision*, *Generalization*, and *Simplicity* [12].

The Fitness dimension represents the ability of a process model to reproduce the control-flow of the traces recorded in the event log [11]. Measuring Fitness can be performed using several approaches such as the "Alignment-based Replay Fitness" [13] and the Petri-net replay technique that allows to detect possible mismatches [9]. The Precision dimension is used to quantify the extra behaviour allowed by a process model that is not recorded in the log [11]. In case the process model contains loops, this will generate infinitely many behaviours. Therefore, counting the number of all possible traces is impossible. There exist several approach to quantify precision such as the *Escaping edges* technique [3], and alignment based technique proposed in [1]. The *generalization* dimension is used to quantify the extent to which the process model can replay log traces that are not yet recorded in the event log [11]. There exist several approaches to estimate generalization such as *the frequency of use* approach, which is based on the assumption that a generic model is a model whose parts are all used with the same frequency [3]. The Simplicity dimension is used to quantify the extent to which a process model is simple. It is defined according to two main principles: (a) The the Occam's Razor principle which states "One should not increase, beyond what is necessary, the number of entities required to explain anything." (b) The understandability of the process model by the user [3]. The literature presents several heuristics that could be used to estimate simplicity such as the *Simplicity by activity occurrence* [3]. This heuristic assumes that the less duplicate activities there are in a process model, the more simple it is.

3.3 Assumptions

The ICI approach is built upon few assumptions. Specifically, the case id is assumed to be explicitly mentioned in the event log. In other words, given a raw event $\hat{e} = (a, t, d_1, \dots, d_k)$ (Definition 3), a case id c is one of the raw event attributes $d_1, \dots, d_k$. Additionally, the case id is given by the same attribute of all raw events. In other words, if d_i is the case id attribute of raw event $\hat{e} \in \hat{\mathcal{E}}$, then the case id attribute of all other events in $\hat{\mathcal{E}}$ is d_i. Finally, the event name attribute and the timestamp attribute of the event log are know and each raw event set refers to executions of the same process (with several instances).

Please note that all these assumptions are typically acceptable in many real process mining projects.

3.4 Approach

The preliminaries presented in Section 3.1, the four quality dimensions described in Section 3.2, and the assumptions presented in Section 3.3 provide a good starting point to describe the ICI approach. The control-flow discovery allows discovering process models reflecting the behaviours seen in an event log [11, p. 125]. To ensure the consistency of the discovered model, the *case id* should be correctly identified in the log. In case it is wrongly identified, the obtained model would be inconsistent. For instance, by selecting a random attribute as a case id it is most likely that the discovered control-flow would not represent the original process model. Consequently, the discovery algorithm used will produce some strange behaviours resulting in an inconsistent model. Luckily, the control-flow four quality dimensions allow quantifying those behaviours.

In principle, the control-flow discovery quality dimensions are meant to evaluate and compare the quality of different discovery algorithms [3]. This paper goes beyond the classical use of the control-flow discovery quality dimensions by exploiting their ability to evaluate and compare different process models obtained using the *same* process discovery algorithm but considering different event attributes as case id.

To Infer the case id attribute from a log file, the ICI approach relies on two important steps. First it uses a heuristic to quantify the number of distinct values of each attribute across the log. This step is called *Compute grouping ratio for* *each attribute*. Its purpose is to identify the attributes that are most likely to represent the case id. In the second step, each log attribute is assumed to be the case id attribute and then evaluated using the control-flow discovery quality dimensions. This step is called *Compute the quality score for each attribute*. Using the heuristic from the first step and the evaluation metrics from the second step, the ICI approach allows to infer the real case id attribute. The remaining of this section (Section 3.4) explains these two steps.

Compute Grouping Ratio for Each Attribute: To get close insight into which event attribute is more likely to represent the case id, the ICI approach computes the *grouping ratio* for each attribute. This is used to quantify the extent to which an attribute can be used to split events into groups, such that each group can be identified by a unique value of the attribute. For instance, a case id attribute is used to group events belonging to the same process execution by assigning them the same value, thus by grouping events by case id, the obtained number of groups will correspond to the number of process executions (cases). However, by choosing a different attribute to represent the case id (i.e., timestamp, event id, resource), the obtained event groups might have smaller or larger size. Section 3.5 provides an example showing the intuition behind the use of the grouping ratio to measure the likelihood that an attribute is the case id.

Let $\hat{L}$ be a set of raw events such that $\hat{L} \subseteq \hat{\mathcal{E}}$. According to Definition 3, a raw event $\hat{e} \in \hat{L}$ is a tuple $\hat{e} = (a, t, d_1, \ldots, d_k)$, with $d_1, \ldots, d_k$ being the list of raw event attributes, where $\forall\, 1 \leq i \leq k, d_i \in \mathcal{D}_i$. $\mathcal{N}_i$ is defined as the set of unique values for $\mathcal{D}_i$ such that $\mathcal{N}_i(\hat{L}) = \{\pi_{d_i}(\hat{e}) \mid \hat{e} \in \hat{L}\}$. Then, the *Grouping Ratio* for $\mathcal{D}_i$ on the raw events set $\hat{L}$ is defined as:

$$g_i(\hat{L}) = 1 - \frac{|\mathcal{N}_i(\hat{L})|}{|\hat{L}|}. \tag{1}$$

Where $|\mathcal{N}_i(\hat{L})|$ is the size of set $\mathcal{N}_i(\hat{L})$, and $|\hat{L}|$ is the size of the set $\hat{L}$ that is the total number of raw events it contains.

Compute Quality Score for Each Attribute: In this step, *raw events* are transformed into *events* with known case id that is one of the event attributes, and then an event log L is generated. Afterwards, a process discovery algorithm is applied to the event log L to discover the corresponding process model. Once the model is obtained, the quality dimension metrics are used to measure fitness, precision, generalization, and simplicity. The measurements are summed up together with the *the distance to the average grouping ratio* then averaged to get a *quality score*, which is used to rank the process model corresponding to each attribute. Finally, the attribute with the highest rank is selected to be the real case id attribute.

Let $\hat{\mathcal{L}}$ be the set of raw events (cf. Definition 3) and let L be an event log (cf. Definition 5). $\hat{\mathcal{L}}$ can be transformed to L as follows: Let k be the number of the of additional data attributes $\mathcal{D}_1, \dots, \mathcal{D}_k$ and let $j \in [1, k]$ be the index of the attribute considered as case id. Then, for each $\hat{e} \in \hat{\mathcal{L}}$, a new event e is created such that $\pi_a(e) = \pi_a(\hat{e})$, $\pi_t(e) = \pi_t(\hat{e})$, $\pi_{d_i}(e) = \pi_{d_i}(\hat{e}), \forall\, 1 \leq i \leq k$, and $\pi_c(e) = \pi_{d_j}(\hat{e})$. Finally e is inserted into the event log L. Also, Let $g(\hat{\mathcal{L}})$ be the average grouping ratio of the of additional data attributes $\mathcal{D}_1, \dots \mathcal{D}_k$ such that $g(\hat{\mathcal{L}}) = \frac{\sum_{i \in [1,k]} g_i(\hat{\mathcal{L}})}{k}$. Then the *distance to the average grouping ratio* for an additional data attribute $\mathcal{D}_i$ is defined as $gr(\hat{\mathcal{L}}, i) = 1 - |g_i(\hat{\mathcal{L}}) - g(\hat{\mathcal{L}})|$, where $|.|$ is an absolute value function.

Algorithm 1 describes the function used to infer the case id. The function takes as input $\hat{\mathcal{L}}$ the set of raw events, and returns c the index of the case id in the list of raw event attributes. The algorithm iterates over all the indexes of the additional attributes (lines 3-16). For each index $i \in [1, k]$, it computes the distance to the average grouping ratio gr using the function $\mathsf{gr}(\hat{\mathcal{L}}, i)$ (line 4), then generates an event log file L using the function $\mathsf{generateLog}(\hat{\mathcal{L}}, i)$ where D_i is assumed to be the case id attribute (line 5), and applies a process discovery algorithm (i.e., Inductive Miner) to generate the corresponding process model M using the function $\mathsf{mine}(L)$ (line 6). Afterwards, it computes the fitness, precision, generalization, and simplicity using the functions $\mathsf{fitness}(M, L)$, $\mathsf{precision}(M, L)$, $\mathsf{generalization}(M, L)$, and $\mathsf{simplicity}(M, L)$ respectively (lines 7-10). Finally it computes the quality score *qual* from the previous quality dimensions and the distance to the average grouping ratio (line 11). Moreover, the algorithm keeps track of the attribute index with highest quality score to return it by the end of all the iterations (lines 12-15).

3.5 Running Example

To illustrate the ICI approach a synthetic log file entitled *Robot Process*[2] recording the workflow of a robot process in a smart factory is used. The important attributes in this log file are the *Case Id*, the *Start Timestamp*, the *Event Name*, and the *Subject Id* that refers to the resource attribute. To demonstrate the ICI approach, the *Case Id* which is the ground truth in this example is assumed to be unknown, and the aim is to infer it as explained in Section 3.4. The first step is to use Equation 1 to calculate the grouping ratio of each event attribute (log column) in the log file. The results are shown in Table 1.

The grouping ratios presented in Table 1 provide a brief insight into which event attributes might represent the case id. By analyzing the obtained grouping ratios, one can notice the following: *Event Id* attribute has score 0, which is trivial since the event Id is a unique identifier for each event. *Start Timestamp* and *End Timestamp* have very low grouping ratios because some events happened concurrently. However, *Event Name*, *Event type*, *Subject Group*, and *Object Group* have very high grouping ratios. Assuming that a case id should not group too many nor too few events one would expect that only *Case Id*, *Subject Id*, and *Object Id* might represent the real case id column.

[2] See `https://doi.org/10.5281/zenodo.1186684`

Algorithm 1: Infer case id

```
Input  : $\hat{\mathcal{L}}$ the set of raw events, where $\mathcal{D}_1, \ldots, \mathcal{D}_k$ are all additional data
         attributes available
Output: $c$ the index of the case id in the list of raw event attributes
1  $best \leftarrow 0$                                   // Initialize highest score
2  $c \leftarrow \perp$
   // Iterate over all the indexes of the additional attributes
      $\mathcal{D}_1, \ldots, \mathcal{D}_k$
3  foreach $i \in [1, k]$ do
4      $gr \leftarrow \mathsf{gr}(\hat{\mathcal{L}}, i)$  // Compute the distance to the average grouping ratio
5      $L \leftarrow \mathsf{generateLog}(\hat{\mathcal{L}}, i)$        // Generate log file using $D_i$ as case id
6      $M \leftarrow \mathsf{mine}(L)$                   // Apply process discovery algorithm
       // Compute all quality dimensions
7      $f \leftarrow \mathsf{fitness}(M, L)$
8      $p \leftarrow \mathsf{precision}(M, L)$
9      $g \leftarrow \mathsf{generalization}(M, L)$
10     $s \leftarrow \mathsf{simplicity}(M, L)$
11     $qual \leftarrow (gr + f + p + g + s)/5$          // Compute quality score
12     if $qual > best$ then
13         $c \leftarrow j$  // Consider $j$ as the index of the new candidate case id
14         $best \leftarrow qual$
15     end
16 end
17 return $c$
```

Table 1. Grouping ratios for *Robot Process* log attributes

CaseId	EventId	StartT	EndT	E.Name	E.Type	S.Group	S.Id	O.Group	O.Id
0.7693	0.0000	0.0079	0.1696	0.9992	0.9997	0.9997	**0.9861**	0.9997	**0.9333**

The second step is to compute the quality score for the attributes in the set of raw events as shown in Table 2. In this step, each attribute is considered as candidate case id. For the sake of simplicity it is assumed that the event name attribute and the start timestamp attribute are known; thus they are excluded from the set of possible attributes. For each candidate case id attribute, the corresponding process model is generated using a process discovery algorithm. In this paper, the "Inductive Miner with Infrequent and all operators (IMfa)" [7] is used. IMfa is considered as one of the least biased discovery algorithms toward the four quality dimensions. Then, the distance to the average grouping ratio *(Gr)* is calculated from the grouping ratios *(G)*. Afterwards, the control-flow discovery quality dimension metrics *(Fr, Pi, Sm, and Gv)* are computed for each attribute and the quality scores *(Quality)* are derived. In this paper, the control-flow discovery quality dimensions are calculated using the metrics presented in [3]: the fitness, the precision, the generalization and the simplicity

Table 2. Quality scores for each candidate case id

Attribute	**G**	**Gr**	**Fr**	**Pi**	**Sm**	**Gv**	**Quality**	**Rank**
Case Id	0.7693	0.9172	0.9998	1.0000	1.0000	0.9726	**0.9779**	**1**
Event Id	0.0000	0.3136	1.0000	1.0000	1.0000	0.9747	0.8577	5
End Timesstamp	0.1696	0.4832	0.9543	0.9171	1.0000	0.9767	0.8663	4
Event Type	0.9997	0.6867	0.7975	0.4927	1.0000	0.8450	0.7644	6
Subject Group	0.9997	0.6867	0.7975	0.4927	1.0000	0.8450	0.7644	6
Subject Id	0.9861	0.7004	0.9997	0.8077	1.0000	0.9596	0.8935	2
Object Group	0.9997	0.6867	0.7975	0.4927	1.0000	0.8450	0.7644	6
Object Id	0.9333	0.7532	0.9991	0.7343	1.0000	0.9738	0.8921	3

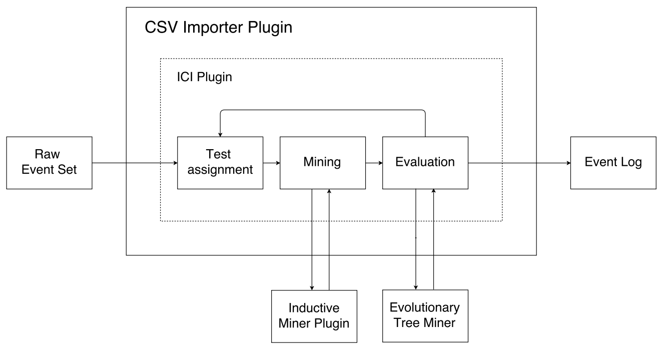


Fig. 2. ICI plugin Architecture

are calculated using *Alignment-based Replay Fitness*, *Escaping edges*, *Frequency of use*, and *Simplicity by activity occurrence* metrics respectively.

By ranking the obtained quality scores shown in Table 2, the *Case Id* attribute represents the best candidate as real case id because it has the highest quality score. This example illustrates the ICI approach and demonstrates its ability to provide accurate results on a synthetic log file. The next section evaluates the ICI approach on real-world event log files.

4 Implementation

An overview of the ICI plugin architecture is depicted in Figure 2. The ICI plugin requires as input a *Raw Event Set* in the CSV format, and an initial mapping with the timestamp and event name attributes. The aim is to infer the case id attribute among the log attributes using the approach introduced in Section 3. For this purpose, the followings three components have been implemented: *(a) Test assignment* which iterates over the log attributes, and constructs an event log where the case id corresponds to one of the log attributes (cf. Algorithm 1, Line 5); *(b) Mining* which provides the event log to a mining algorithm and returns the corresponding model (cf. Algorithm 1, Line 6); *(c) Evaluation* which evaluates the model using the control-flow discovery quality dimensions (cf. Algorithm 1, Lines 7-10). By end of this process, the event log with the highest quality score is chosen.

The ICI plugin integrates two main packages from the open-source process mining framework ProM[3] that are the *Inductive Miner*[4] which implements the IMfa algorithm, and the *Evolutionary Tree Miner*[5] which implements the necessary metrics used to evaluate the control-flow four quality dimensions. The ICI plugin is embedded with the CSV Importer Plugin[6].

As mentioned in Section 3.4 choosing the wrong attribute to represent the case id causes inconsistencies with the mining algorithm. These inconsistencies are amplified when choosing an attribute with too low or too high grouping ratio. As solution, the ICI plugin allows choosing a sample of the event log instead of using the full event log to infer the case id, which reduces the plugin execution time and avoids memory overheads due to the inconsistencies in the mining algorithm. A sample can be selected from the top n entries of the log or from a random selection of n entries in the log.

The ICI plugin (called "Infer Case ID" in ProM) is available as part of the CSV Importer package[7]. It can be installed using the ProM Package Manager. A video demonstration of the plugin is available at `https://youtu.be/OKyuc3mEG1I`.

5 Evaluation

To evaluate the ICI approach on a larger scale, several real-world event logs were obtained from the 4TU public database[8]. With the purpose of having a ground truth to evaluate the accuracy of ICI approach, the event logs used are all *labelled*: the real case id attribute is known. This section reports the results obtained by applying the ICI approach on several real-world log files. Section 5.1 explains the evaluation procedure and presents the used data sets, and Section 5.2 reports the evaluation results.

5.1 Evaluation Procedure and Data Sets

Most of the event logs obtained from the 4TU database are available in XES format, thus, the attribute labels are already known. The process mining tool Disco[9] was used to convert the event logs from XES to CSV format. The sample

[3] See `http://www.promtools.org/`
[4] See `https://svn.win.tue.nl/repos/prom/Packages/InductiveMiner/`
[5] See `https://svn.win.tue.nl/repos/prom/Packages/EvolutionaryTreeMiner/`
[6] See `https://svn.win.tue.nl/repos/prom/Packages/CSVImporter/`
[7] Currently available in ProM Nightly Build at `http://www.promtools.org/doku.php?id=nightly`
[8] See the collection of real-world event logs at 4TU Center for Research Data `http://data.4tu.nl/repository/collection:event_logs_real`
[9] See `https://fluxicon.com/disco/`

used in the evaluation consists of the top 10000 events ordered by timestamp (in an ascending order) for each event log. The plugin was executed with the following system configuration: 12Gb of RAM, and 1 processor with 4 cores.

The data sets considered to evaluate the ICI approach are the following: BPI challenge 2012, BPI challenge 2013 incidents, BPI challenge 2014 Detail Incident Activity, BPI challanges 2017, Credit requirements, Helpdesk anonymized[10] and Receipt phase of an environmental permit application process (WABO) CoSeLoG project. These event logs are available in the Real Event Logs collection[11] of the 4TU database.

5.2 Results

The evaluation results are reported in Table 3. For each log file event attribute the following ratios are computed: grouping ratio for the full log file *(G full)*, grouping ratio of the sample used in the evaluation *(G sample)*, distance to average grouping ratio of the sample *(Gr)*, alignment-based replay fitness *(Fr)*, precision using escaping edges improved technique *(Pi)*, generalization using frequency of use technique *(Gv)*, simplicity using activity occurrence technique *(Sm)* and quality score *(Quality S.)*. The obtained quality scores are ranked to infer the log attribute with the highest rank. Note that the quality score calculation considers the grouping ratio of the sample instead of the grouping ratio of the full log. However, both grouping ratios (sample and full) are reported to emphasis on the fact that the grouping ratio of the sample used does not differ much from the grouping ratio of the original log file.

To demonstrate the accuracy of the ICI approach, the real case id attribute (considered as ground truth) should always have the highest rank among the other attributes. For sake of brevity, only the top three attributes with the highest rank are reported in Table 3. The evaluation results show that the case id attribute always has the highest quality score. The results are discussed in details in Section 6. The complete version of the evaluation results including the quality scores of all the attributes considered for each event log is available at `https://doi.org/10.5281/zenodo.1186678`.

6 Discussion

This section discusses the evaluation of the ICI approach based on the data shown in Table 3. Clearly, the ICI approach demonstrates a high accuracy in inferring the case id in all the event logs considered for the evaluation. However, it is still important to highlight few cases where the quality score of other event attributes is very close to the case id attribute score. For instance, in BPI challenge 2013, the quality scores for *Case ID* attribute and *Resource* attribute are 0.8834 and 0.8745 respectively. To explain this small difference in the quality scores,

[10] See `https://data.mendeley.com/datasets/nm9xkzhpm4/1`

[11] See `http://data.4tu.nl/repository/collection:event_logs_real`

Table 3. Quality scores of the three top ranked attributes of each event log. (refer to Section 5.1 for the full names the event logs.)

Log file	Attribute	G full	G sample	Gr	Fr	Pi	Sm	Gv	Quality S.	Rank
	Case ID	0.9501	0.9210	0.9267	0.9097	0.7346	1.0000	0.9170	**0.8976**	1
BPI challenge 2012	(case) AMOUNT_REQ	0.9976	0.9881	0.8596	0.9998	0.4074	1.0000	0.9076	0.8349	2
	concept:name	0.9999	0.9976	0.8501	0.9109	0.6615	1.0000	0.5169	0.7879	3
	Case ID	0.8847	0.9045	0.9993	0.9926	0.4574	1.0000	0.9678	**0.8834**	1
BPI challenge 2013	Resource	0.9780	0.9223	0.9815	0.9910	0.4358	1.0000	0.9640	0.8745	2
	impact	0.9999	0.9996	0.9042	0.9588	0.6001	1.0000	0.8457	0.8618	3
	Incident ID	0.9471	0.9471	0.8049	0.9977	0.3541	1.0000	0.8922	**0.8098**	1
BPI challenge 2014	IncidentActivity_Number	0.0000	0.0000	0.2480	1.0000	1.0000	1.0000	0.7271	0.7950	2
	Assignment Group	0.9886	0.9886	0.7634	0.9861	0.2568	1.0000	0.8846	0.7782	3
	Case ID	0.9439	0.9084	0.9246	0.9806	0.7591	1.0000	0.9370	**0.9203**	1
BPI challanges 2017	(case) RequestedAmount	0.9988	0.9901	0.8429	0.9858	0.5070	1.0000	0.9489	0.8569	2
	EventID	0.0000	0.0912	0.2582	1.0000	1.0000	1.0000	0.9135	0.8343	3
	Case ID	0.8750	0.8712	0.7327	0.9916	1.0000	1.0000	0.9718	**0.9392**	1
Credit requirements	Complete Timestamp	0.0127	0.0123	0.4084	0.9983	0.8900	1.0000	0.9821	0.8558	2
	Resource	0.9999	0.9992	0.6047	0.9442	0.6677	1.0000	0.4905	0.7414	3
	Case ID	0.8168	0.8180	0.8870	0.9643	0.9729	1.0000	0.9494	**0.9547**	1
Helpdesk anonymized	customer	0.9832	0.9727	0.9583	0.9302	0.5364	1.0000	0.8786	0.8607	2
	product	0.9990	0.9984	0.9326	0.9401	0.4979	1.0000	0.8761	0.8493	3
	Case ID	0.8327	0.8327	0.9850	0.9843	0.7017	1.0000	0.8838	**0.9110**	1
Receipt env. Permit	(case) responsible	0.9953	0.9953	0.8223	0.9967	0.2582	1.0000	0.8714	0.7897	2
	org:group	0.9987	0.9987	0.8190	0.9795	0.3507	1.0000	0.7992	0.7897	3

Table 4. Quality scores for BPI challenge 2013 with a sample size of 40000 events

Log file	Attribute	G sample	Gr	Fr	Pi	Sm	Gv	Quality	Rank
	Case ID	0.8666	0.9624	0.9641	0.8205	1.0000	0.9480	**0.9390**	1
BPI Chal. 2013 (40000 events)	Resource	0.9674	0.9368	0.9001	0.4056	1.0000	0.8938	0.8273	2
	product	0.9846	0.9196	0.9996	0.2387	1.0000	0.9709	0.8258	3

the process mining tool Disco was used. By inspecting the statistics provided by the tool for the process model where the case id corresponds the real case id attribute, the number of cases and resources are 954 and 776 respectively, which can also be noticed from the grouping ratios of *Case ID* attribute and *Resource* attribute that are 0.9045 and 0.9223 respectively. This small difference in the quality score can be explained with the fact that the sample of the event log used considers only the top 10000 events which is not enough to perceive the overall behaviour of the model. Alternatively, a large sample of 40000 events is applied to the BPI challenge 2013 event log. The corresponding quality scores are shown in Table 4.

As shown in Table 4, by increasing the event log sample size, the difference between the quality scores for *Case ID* attribute and *Resource* attribute increased significantly (0.9390 and 0.8273 respectively). In a perfect scenario, one would use the full event log to preserve its overall behaviour. However, memory overhead issues might happen especially while dealing with large event logs.

BPI challenge 2014 represents another example where the difference in quality scores between *Case ID* attribute and *IncidentActivityNumber* attribute is insignificant (0.8098, and 0.7950 respectively). However, the grouping ratio of

IncidentActivityNumber attribute is 0; hence, the attribute contains only unique values. Consequently, the generated model would appear like a flower model where all the log traces can be replayed, thus, the fitness will certainly be equal to 1. [11, p. 151]. To avoid such cases, the attributes with a grouping ratio equals to 0 could be filtered out before running the ICI plugin.

The evaluation results demonstrate that the ICI approach is accurate on several event logs. However, more event logs should be tried-out and other control-flow discovery algorithms should be used. Nevertheless, the results shown in Section 5.2 are promising and provides a clear insight into the aspects that should be enhanced. Mainly, the following challenges should be addressed: *(a)* Filtering out the log attributes with too low or too high grouping ratios. By overcoming this challenge, the memory overhead issues will be avoided. Moreover, the attributes with Boolean values will be ignored. Such attributes always have higher grouping ratio, which impacts negatively on the distance to average grouping ratio in case several log attributes are Booleans. *(b)* Defining an optimal sample size proportional to each event log characteristics (i.e., number of resources). Indeed, The accuracy of the ICI approach depends on the sample of the log used to compute the quality score and its ability to preserve the overall behaviour, which can be quantified by the control-fow discovery quality dimensions and the grouping ratio. Please note that the sample size used in this evaluation has been chosen to preserve the overall behaviour of the log. However, it cannot guarantee that the same sample size is valid for all other log files.

7 Conclusion and Future Work

To sum up, the ICI approach proposes a new technique to automatically infer the case id from an event log by exploring the control-flow discovery quality dimensions capabilities. Unlike the existing techniques mentioned in Section 2, the ICI approach does not require any domain-specific heuristics. Under the assumption that a case id is explicitly mentioned in the event log, the ICI approach allowed to correctly identify the case id in a synthetic log. The approach was evaluated using several real-world event logs obtained from a public database to demonstrate its accuracy on a large scale. The results show a high potential for inferring the case id despite the challenges discussed in Section 6.

As future work, the challenges related to memory overhead and optimal sample size have the highest priority. Moreover. several heuristics could be applied to filter out the candidate event attributes based on their data types (i.e., ignoring timestamp attributes and Boolean attributes) and based on their grouping ratios. In addition, the availability of domain knowledge will help to reduce the search space and enhance the performance of the ICI approach by enabling a pre-selection of the event attributes that are most likely to contain the case id. Furthermore, the ICI approach could be easily generalized to infer the case id from a combination of log attributes. In term of feasibility, the approach could be illustrated in a practical use-case fitting into one of the application areas mentioned in Section 1. Finally, the ICI algorithm could be tried-out using other control-flow discovery algorithms and quality metrics.

References

1. A. Adriansyah, J. Munoz-Gama, J. Carmona, B. F. van Dongen, and W. M. P. van der Aalst. Measuring precision of modeled behavior. *Information Systems and e-Business Management*, 13(1):37–67, feb 2015.
2. D. Bayomie, I. M. Helal, A. Awad, E. Ezat, and A. ElBastawissi. Deducing case IDs for unlabeled event logs. *Lecture Notes in Business Information Processing*, 256:242–254, 2016.
3. J. C. a. M. Buijs. *Flexible Evolutionary Algorithms for Mining Structured Process Models.* 2014.
4. A. Burattin. *Process Mining Techniques in Business Environments*, volume 207 of *Lecture Notes in Business Information Processing.* Springer International Publishing, Cham, 2015.
5. A. Burattin and R. Vigo. A framework for semi-automated process instance discovery from decorative attributes. In *2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, pages 176–183. IEEE, apr 2011.
6. D. R. Ferreira and D. Gillblad. Discovering Process Models from. *Bpm*, pages 143–158, 2009.
7. S. J. J. Leemans. *Robust process mining with guarantees.* SIKS Dissertation Series No. 2017-12, 2017.
8. M. Polato, A. Sperduti, A. Burattin, and M. d. Leoni. Time and activity sequence prediction of business process instances. *Computing*, Feb 2018.
9. A. Rozinat and W. M. P. van der Aalst. Conformance checking of processes based on monitoring real behavior. *Information Systems*, 33(1):64–95, 2008.
10. e. a. Van Der Aalst. Process mining manifesto. *Lecture Notes in Business Information Processing*, 99 LNBIP(PART 1):169–194, 2012.
11. W. M. Van Der Aalst. *Process mining: Discovery, Conformance, and Enhancement of Business processes.* Springer International Publishing, 2010.
12. W. M. Van Der Aalst. Mediating between modeled and observed behavior: The quest for the 'right' process: Keynote. *Proceedings - International Conference on Research Challenges in Information Science*, 2013.
13. W. M. P. van der Aalst. Replaying History on Process Models for Conformance Checking and Performance Analysis. (1):1–18.
14. M. Walicki and D. R. Ferreira. Sequence partitioning for process mining with unlabeled event logs. *Data and Knowledge Engineering*, 70(10):821–841, 2011.
15. M. Weidlich, A. Polyvyanyy, J. Mendling, and M. Weske. Causal behavioural profiles - Efficient computation, applications, and evaluation. *Fundamenta Informaticae*, 113(3-4):399–435, 2011.