Improving Accuracy and Explainability in Event-Case Correlation via Rule Mining

Dina Bayomie
Wirtschaftsuniversität Wien
Vienna, Austria
dbayomie@wu.ac.at

Kate Revoredo
Humboldt-Universität zu Berlin
Berlin, Germany
kate.revoredo@hu-berlin.de

Claudio Di Ciccio
Sapienza University of Rome
Rome, Italy
claudio.diciccio@uniroma1.it

Jan Mendling
Humboldt-Universität zu Berlin
Berlin, Germany
jan.mendling@hu-berlin.de

***Abstract*—Process mining analyzes business processes' behavior and performance using event logs. An essential requirement is that events are grouped in cases representing the execution of process instances. However, logs extracted from different systems or non-process-aware information systems do not map events with unique case identifiers (case IDs). In such settings, the event log needs to be pre-processed to group events into cases – an operation known as event correlation. Existing techniques for correlating events work with different assumptions: some assume the generating processes are acyclic, others require extra domain knowledge such as the relation between the events and event attributes, or heuristic information about the activities' execution time behavior. However, the domain knowledge is not always available or easy to acquire, compromising the quality of the correlated event log. In this paper, we propose a new technique called EC-SA-RM, which correlates the events using a simulated annealing technique and iteratively learns the domain knowledge as a set of association rules. The technique requires a sequence of timestamped events (i.e., the log without case IDs) and a process model describing the underlying business process. At each iteration of the simulated annealing, a possible correlated log is generated. Then, EC-SA-RM uses this correlated log to learn a set of association rules that represent the relationship between the events and the changing behavior over the events' attributes in an understandable way. These rules enrich the input and improve the event correlation process for the next iteration. EC-SA-RM returns an event log in which events are grouped in cases and a set of association rules that explain the correlation over the events. We evaluate our approach using four real-life datasets.**

***Index Terms*—Event correlation, Association rule mining, Simulated annealing, Explainability**

I. Introduction

Process mining techniques [1] are used to analyze the behavior and performance of a variety of processes. A mandatory input of the techniques is an event log composed of cases, wherein each case is a sequence of events. As every event relates with exactly one case, we name this data structure as correlated log. In practice, though, process data is often stored by different systems (also non-process-aware ones), thus a mapping of events to unique cases is not feasible – therefore, such a correlated event log is not available.

The work of J. Mendling was supported by the Einstein Foundation Berlin. The work of C. Di Ciccio was supported by the MUR under the PRIN programme, grant B87G22000450001 (PINPOINT), the "Dipartimenti di eccellenza 2018-2022" grant for the Department of Computer Science of Sapienza, and by the DRONES and SPECTRA Sapienza research projects. This work received funding from the Teaming.AI project in the EU Horizon 2020 research and innovation program under grant agreement No 95740.

To address this issue, several approaches for correlating events were proposed. They work under different assumptions about the domain, such as acyclicity of the process [2], [3], the existence of case identifiers among the event attributes [4], [5], a profiling of the activities' execution time [6], or additional data rules about event attributes [7]. The experiments conducted in [7] highlight that a more detailed domain knowledge (in the form of accurate data rules) implies a higher quality of the output correlated event log. However, extra information about the domain is not always available or easy to acquire.

In this work, we assume that domain knowledge is not given in advance. Our approach learns rules during the event correlation process based on partial versions of the correlated event log. To this end, we build on the event correlation process presented in [7]. This approach uses simulated annealing to iterate over the search space and generate a correlated event log. In our approach, at each iteration of the simulated annealing we use the current correlated event log to learn data rules represented as association rules, and evaluate them. The rules with the highest accuracy are then used as domain knowledge input in the next iteration – until convergence. We employ the technique presented in [8] to learn and evaluate the association rules, which describe the relationship between the events and the changing behavior over the events' attributes, i.e., the way the values of data attributes match or vary depending on the activities enacted in the sequence. We evaluate our approach using four real-life datasets. The results show that our approach is able to learn a correlated event log with better quality, compared with approaches that use no domain knowledge or partial domain knowledge. Furthermore, the extracted association rules provide an explanation for the event correlation that can help the analyst understand how the correlated event log was generated.

The remainder of the paper is structured as follows. Section II describes related work. Section III presents some fundamentals necessary for the understanding of our approach, which is described in Section IV. Section V details the evaluation of our approach. Finally, Section VI concludes our work and presents future directions for the research.

II. Related Work

Several techniques address the event correlation problem. The techniques most related to ours are the ones that resort to some domain knowledge during the correlation process.

2022 4th International Conference on Process Mining (ICPM 2022), ISBN 979-8-3503-9714-7

The Deducing Case Ids (DCIc) approach [6] requires a process model and heuristic information about the activities' execution duration behavior. DCIc uses a breadth-first approach to build a case decision tree and explore the solution space in order to correlate events of an acyclic process. It is sensitive to the quality of the process model. Also, it is computationally inefficient due to the breadth-first search approach.

The Event Correlation by Simulated Annealing (EC-SA) approach [9] uses the event names and timestamp in addition to the process model. EC-SA addresses the correlation problem as a multi-level optimization problem, as it searches for the nearest optimal correlated log considering the fitness with an input process model and the activities' time profile within the log. The accuracy of the given model affects the quality of the correlated log, and the performance depends on the number of uncorrelated events. EC-SA-Data, introduced in [7], extends EC-SA by using the domain knowledge represented as business constraints over the event data attributes.

Motahari-Nezhad et al. [10] propose a semi-automated correlation approach to correlate the web service messages based on correlation conditions. The approach infers the correlation conditions using the event data from different data layers. Also, it computes the interestingness of the attributes of the events to prune the search space. Thus, it generates several log partitions and possible process views. The approach requires user-defined domain parameters and intermediate domain expert feedback to guide the correlation process.

De Murillas et al. [11] provide a way to automatically generate different event logs from a database by defining the case notion based on the data relations in the data model. A case notion specifies the events for the correlation based on the selected data objects representing the investigated cases. They measure the interestingness of the generated logs and recommend the highest one to the user. However, the used log interestingness metrics do not cover key aspects such as the homogeneity of behavior captured in the event log.

Abbad Andaloussi et al. [4] assume the existence of a case identifier within the event attributes. They propose a method that compares the discovered process models by considering each event log attribute as a case identifier. Bala et al. [5] follow a similar direction based on the idea that identifiers are repetitive in the log.

In summary, these recent techniques make assumptions about process behavior, available information and size of search space. The technique presented in this paper learns knowledge about the association between the control-flow and data attributes behavior from cases defined in the course of the correlation process. This way, it relaxes the dependence on prior assumptions, as it resorts to knowledge uncovered from the data.

III. Preliminaries

In this section, we discuss the fundamental notions that our approach builds upon. Section III-A presents the event data structures. Section III-B reviews the rule mining method we use in our solution. Section III-C reviews the event correlation technique, which we integrate at the core of our solution.

A. Process Event Data Structures

Starting with the basic notion of event (i.e., the atomic unit of execution), we introduce the uncorrelated event log, case and event log. An *event* e represents the execution of a process activity. An event is associated with a set of attributes ($\mathfrak{A}$), which provide information about the recorded activity (Act), the timestamp marking the date and time of execution (Ts), and (optionally) additional context information such as resource, cost, etc. Every *attribute* $\mathrm{Attr} \in \mathfrak{A}$ is mapped to one of the attribute's domain values, i.e., an element in the non-empty set $\mathrm{Dom}(\mathrm{Attr})$. We indicate the value mapped by Attr to an event e by using a dot notation (i.e., $e.\mathrm{Attr}$).

For example, e_1 in Fig. 1 is mapped to four different domain values, one per attribute: $e_1.\mathrm{Act} = \mathsf{A}$ represents the executed activity, $e_1.\mathrm{Ts} =$"$07/06/2022\ 09:00$" represents the completion timestamp, $e_1.\mathrm{Res} = \mathsf{Kate}$ represents the operating resource, and $e_1.\mathrm{Type} = \mathsf{Home}$ represents additional data knowledge about the event context.

Definition 1 (Uncorrelated log): An *uncorrelated event log* (or *uncorrelated log* for short) UL is a finite set of events $E \ni e$ with total order defined over E, $\preccurlyeq \subseteq E \times E$.
We assume the mapping of Ts to be coherent with $\preccurlyeq$, i.e., if $e \preccurlyeq e'$ then $e.\mathrm{Ts} \leqslant e'.\mathrm{Ts}$. Considering the total ordering as a mapping from a convex subset of integers, we can assign to every event a unique integer index (or *event id* for short), induced by $\preccurlyeq$ on the events. We shall denote the index $i \in [1, |E|]$ of an event e as a subscript, e_i. For example, Fig. 1(a) depicts an uncorrelated log and e_1 is its first event.

Definition 2 (Case): A *case* $\sigma = \langle e_{\sigma,1}, \ldots, e_{\sigma,m} \rangle$ is a finite sequence of length $m \in \mathbb{N}$ of events $e_{\sigma,i}$ with $1 \leqslant i \leqslant m$ induced by $\preccurlyeq$, i.e., such that $e_{\sigma,i} \preccurlyeq e_{\sigma,k}$ for every $i \leqslant k \leqslant m$.
We assume every case to be assigned a unique case identifier (case id for short), namely an integer in a convex subset.
We shall denote the i-th event within a case σ as $\sigma(i)$. In our example, the first event in σ_1 is denoted as $\sigma_1(1)$, whereby $\sigma_1(1) = e_1$.

Definition 3 (Correlated Event log): A *correlated event log* (or *event log* for short) $L = \{\sigma_1, \ldots, \sigma_n\}$ is a finite non-empty set of cases, such that if $e \in \sigma_i$, then $e \notin \sigma_j$ for all $i, j \in [1..n]$, $i \neq j$. We denote its cardinality $n \in \mathbb{N}$ as $|L|$.
We shall refer to a case σ that contains an event e in an event log L with $L(e)$. For example, Fig. 1(d) depicts an event log that contains 3 cases. Case σ_1 defined by case id 1 is $\langle e_1, e_2, e_6 \rangle$. We write $L(e_6) = \sigma_1$. Notice that it preserves the order of the events within the case.

B. Event Log Rule Miner

The Event Log Rule Miner method [8], henceforth called EL-RM, discovers association rules between the control-flow and the data objects within an event log. It is inspired by the knowledge discovery in database (KDD) process [12] and proceeds in four main steps, as illustrated in Fig. 2: *(i)* preparing the event log, *(ii)* transforming the event log, *(iii)* mining the

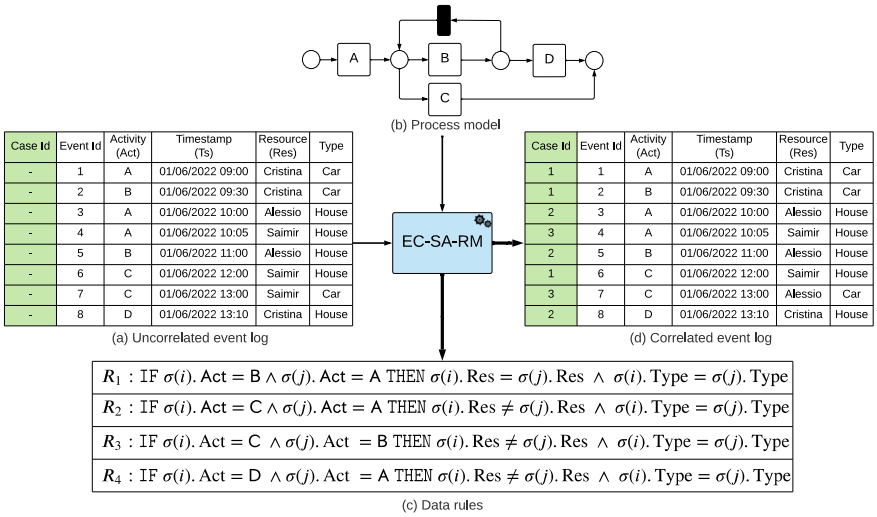


Figure 1: Running example of a sample loan application check process

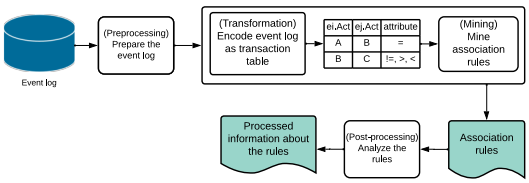


Figure 2: The EL-RM method [8]

event log, and *(iv)* post-processing the discovered association rules. In the following, we will refer to the latter as *data rules* or simply *rules* for short.

The first step prepares the event log based on the process analyst's objectives, such as filtering cases or partitioning the event log. The second step encodes the pre-processed event log as a transaction table that sustains the control-flow and data perspectives. In the third step, the EL-RM is applied on the transaction table to discover the rules. In the fourth step, the rules are post-processed based on the process analyst's objectives such as ranking them using the interestingness measures of support, confidence and lift [13], or combining the rules.

EL-RM extracts rules in the following form (interpreting $\wedge$ as the logical conjunction, $<, >, =, \neq$ as as comparison operators over domain values, $a, b \in$ Dom(Attr), and i, j as positive integers):

$$
\begin{aligned}
R &:= \texttt{IF}\ R_{\texttt{IF}}\ \texttt{THEN}\ R_{\texttt{THEN}} \qquad (1)\\
R_{\texttt{IF}} &:= e_i.\mathrm{Act} = a \ \wedge\ e_j.\mathrm{Act} = b\\
R_{\texttt{THEN}} &:= e_i.\mathrm{Attr} \lesseqgtr e_j.\mathrm{Attr}'\\
\lesseqgtr &:= < \mid > \mid = \mid \neq
\end{aligned}
$$

In the following, we shall name $R_{\texttt{IF}}$ and $R_{\texttt{THEN}}$ *antecedent* and *consequent* of the rule, respectively.

C. Event correlation with Simulated Annealing

Simulated Annealing (SA) is a meta-heuristic optimization algorithm that searches the solution space for the nearest

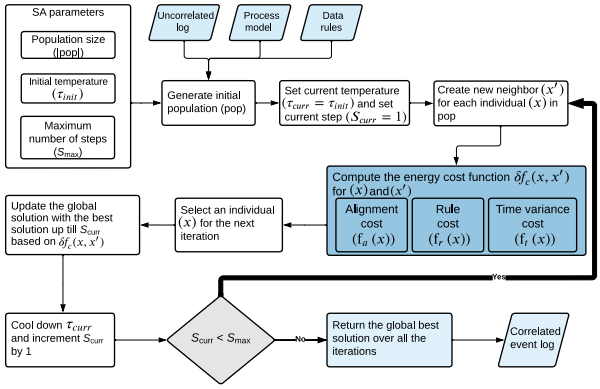


Figure 3: The EC-SA-Data technique overview [7]

global solution. It simulates metals' cooling through the annealing process [14]: during the earlier iterations, the so-called *temperature* is higher and candidate solutions (named *individuals*) from wider areas of the search space are picked; at every iteration, the temperature cools down and the explored search space shrinks, getting closer to the previous solution. EC-SA-Data [7] resorts to SA to solve the correlation problem. Using SA helps find an *approximate* global optimal correlated log in a reasonable time.

Figure 3 shows an overview of EC-SA-Data. It requires three inputs: (1) an uncorrelated log (UL), (2) a process model (PM), and (3) a set of domain knowledge rules, i.e., data rules on the event data attributes $\{R_1, \ldots, R_m\}$. It generates a correlated event log (L) as an output. Next to the input above, SA (hence EC-SA-Data) allows the user to influence the annealing process with the following parameters: (1) the initial temperature ($\tau_{\text{init}} \geq 1$), (2) the maximum number of steps ($S_{\max} \geq 1$), and (3) the number of individuals to generate at each iteration (namely the *population*, $|\text{pop}| \geq 1$). The analysts can change these parameters based on their experience.

The event correlation problem is treated by EC-SA-Data as

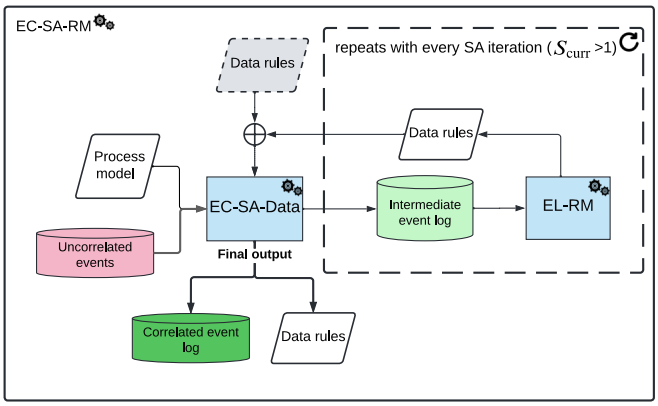


Figure 4: Overview of EC-SA-RM

a multi-level optimization problem with three nested objectives: (1) minimizing the misalignment between L and PM, (2) minimizing the violations of rules over cases in L, (3) minimizing the activity execution time variance across the cases in L.

EC-SA-Data proceeds through the following steps. First, it creates the initial population. Then, it initializes the current step counter ($S_{\text{curr}} = 1$) and the current temperature with an initial temperature, $\tau_{\text{curr}} = \tau_{\text{init}}$. Afterwards, the iterative annealing process begins by generating a neighbor solution x' that uses the current individual x. Next, it computes the energy cost function $\delta \text{f}_\text{c}(x, x')$ between x and x' based on three energy functions. The first energy function ($\text{f}_\text{a}(x)$) computes the cost of aligning x and PM. The second energy function ($\text{f}_\text{r}(x)$) computes the data rules violations cost within x. The third energy function ($\text{f}_\text{t}(x)$) computes the activity execution time variance within x. Then SA computes the acceptance probability $\text{prob}(x')$ using $\delta \text{f}_\text{c}(x, x')$. $\text{prob}(x')$ is used to determine if the new neighbor, x', can be used for the next iteration as a reference individual, even if it may not perform better than x. Based on this, SA can increase the chances of skipping the local optimum and let the algorithm explore the search space further. Finally, SA uses a cooling schedule that defines the rate at which the temperature (τ_{curr}) cools down. Finally, it increments S_{curr} by 1. SA repeats the annealing and cooling process till S_{curr} reaches the maximum number of iterations ($S_{\max}$).

EC-SA-Data creates an individual x by correlating every uncorrelated event $e \in \text{UL}$. For every e an event correlation decision is taken based on two stages. First, it replays the running cases on the process model to filter out the candidate assignments for e. Then, it uses domain rules to rank the candidate cases based on the number of satisfied rules by e in those cases.

Notice that EC-SA-Data is highly sensitive to the process model and the domain rules. If the domain rules are not available or incomplete, it compromises the quality of the generated log.

IV. The EC-SA-RM Solution

Figure 1 illustrates EC-SA-RM with a running example. The EC-SA-RM technique requires the following input: (a) an uncorrelated log (UL), and (b) a process model (PM). It can also receive a set of data rules as an optional third input element. As an output, EC-SA-RM generates (c) a correlated event log (L), and (d) a set of data rules ($\mathfrak{R}$) that explain the event correlation within the log.

The event correlation problem is treated by EC-SA-RM as a multi-level optimization problem endowed with a learning capability. EC-SA-RM employs EC-SA-Data to assign the events with a case (henceforth, *correlate* events) following the simulated annealing iterations (see Section III-C). However, for each iteration, EC-SA-RM uses EL-RM to discover new data rules using the selected individual x as an intermediate event log. In the next iteration, the learned data rules are given as input. From this passage, EC-SA-RM allows SA to explore the solution space with more information to improve the next individual x'. Figure 4 depicts a detailed schema of EC-SA-RM. We describe the steps in detail in the following subsections.

A. Event Correlation Decision

An event e should be assigned with a correlated case σ in every individual x. However, each event has multiple candidate cases. We denote with P_e the set of possible cases for e. The higher the cardinality of P_e is, the more randomized the decision on the assignment (henceforth, *correlation decision*) gets. We formalize this notion as follows.

Definition 4 (Randomization Factor): Let $e \in \text{UL}$ be an uncorrelated event, and $P_e \in 2^L$ a set of cases that e can be correlated with. The function $\text{Rand} : E \times 2^L \rightarrow [0, 1]$ computes the randomization factor of an event e given the possible cases P_e as follows:

$$\text{Rand}(e, P_e) = 1 - \frac{1}{|P_e|} \tag{2}$$

In the following, we omit the term P_e whenever clear from the context and denote the randomization factor of an event e with the dot notation to emphasize its use as a meta-attribute in our approach. If $e.\text{Rand} = 0$, then e the correlation decision is made without uncertainty (only one case can be assigned).

To find the candidate cases for each event in the uncorrelated log, EC-SA-RM filters out the cases in two steps. The first step prunes the possible cases based on the model. A case represents a replay of the process model from the start activity (A in Fig. 2) to one of the end activities (D in Fig. 2). We name the cases that do not reach the end of the process model as *open cases*. There are three scenarios when replaying an event e over the input process model:

1) Event e corresponds to the execution of the start activity of the process model (we name it *start event*). Then, a new case is open. Notice that the start event has no randomization factor ($e.\text{Rand} = 0$).
2) Event e corresponds to the execution of an enabled (non-start) activity for one or more cases in P_e. We call e an *enabled event* in this case. If only one case $\sigma \in P_e$ enables e.Act, it is assigned with σ and thus $e.\text{Rand} = 0$. Otherwise, e is assigned with the case satisfying the highest number of data rules in the next step.

Case Id	Event Id	Activity (Act)	Timestamp (Ts)	Resource (Res)	Type	Rand
1	1	A	01/06/2022 09:00	Cristina	Car	0
1	2	B	01/06/2022 09:30	Cristina	Car	0
2	3	A	01/06/2022 10:00	Alessio	House	0
3	4	A	01/06/2022 10:05	Saimir	House	0
2	5	B	01/06/2022 11:00	Alessio	House	0.67
1	6	C	01/06/2022 12:00	Saimir	House	0.67
2	7	C	01/06/2022 13:00	Saimir	Car	0.5
3	8	D	01/06/2022 13:10	Cristina	House	0.67

(a) First iteration, individual x_1

R_1 : IF $\sigma(i)$. Act = B $\wedge$ $\sigma(j)$. Act = A THEN $\sigma(i)$. Res = $\sigma(j)$. Res
R_2 : IF $\sigma(i)$. Act = B $\wedge$ $\sigma(j)$. Act = A THEN $\sigma(i)$. Type = $\sigma(j)$. Type

(b) Discovered rules over x_1

R_1 : IF $\sigma(i)$. Act = B $\wedge$ $\sigma(j)$. Act = A THEN $\sigma(i)$. Res = $\sigma(j)$. Res $\wedge$ $\sigma(i)$. Type = $\sigma(j)$. Type

(c) Combined rules over x_1

Figure 5: Running example, given the UL in Fig. 1(a), PM in Fig. 1(b) and no initial input rules

3) Event e does not correspond to any enabled activity (*non-enabled event*). Then, all the open cases are considered as possible assignments for the next step.

The second step ranks the possible candidate cases P_e based on the number of data rules that are satisfied by e in those cases. When exactly one case achieves the highest number of satisfied rules, then $e.\mathrm{Rand} = 0$ as there is no randomization factor in assigning this event based on the given domain knowledge. Otherwise, the randomization factor is computed based on the number of highest ranking cases.

For example, let us consider UL in Fig. 1(a) and PM in Fig. 1(b). In the first iteration, there are no input rules, i.e., $\mathfrak{R} = \emptyset$. The events are correlated based on the model replay only, as shown in Fig. 5(a). Event e_1 is a start event as it executes the start activity (A). Thus, it opens a new case (σ_1) and sets the randomization factor to zero: $e_1.\mathrm{Rand} = 0$. The same goes for e_3 and e_4, which start σ_2 and σ_3, respectively. Then, σ_1 is the only open case within the log before e_2 and it expects the execution of activity B. Thus, e_2 is assigned to σ_1 and $e_2.\mathrm{Rand} = 0$. There are three open cases in the uncorrelated log before e_5: σ_1, σ_2 and σ_3 expect the execution of activity B. Therefore, e_5 is an enabled event and the three cases are considered as possible candidate cases: $P_{e_5} = \{\sigma_1, \sigma_2, \sigma_3\}$. Thus, e_5 has randomization factor $e_5.\mathrm{Rand} = 0.67$. On the other hand, none of the cases σ_1, σ_2 and σ_3 expect the execution of activity D. Consequently, e_8 is a non-enabled event and all the three cases are considered as possible candidate cases: $P_{e_8} = \{\sigma_1, \sigma_2, \sigma_3\}$. Therefore, e_8 has randomization factor $e_3.\mathrm{Rand} = 0.67$.

B. Applying EL-RM

EC-SA-RM uses EL-RM to learn new data rules that can improve the event correlation decision and reduce the randomization factor over the SA iterations. Figure 2 shows the steps of the EL-RM method introduced in Section III-B.

First, we have a pre-processing step in which EC-SA-RM uses the selected individual x as an intermediate correlated log. It filters cases that contain at least two events (i.e., at least one more than the start event with a randomization factor equal to zero) and selects only these events to represent the case in the analysis. For example, let us consider the event log L in Fig. 5(a). Within the pre-processing step, we filter out only two events, namely e_1 and e_2, because they belong to the same case (σ_1) and $e_1.\mathrm{Rand} = e_2.\mathrm{Rand} = 0$. However, notice that during the first iterations there may be no events with a randomization factor equal to zero but the initial ones. In case, we take into account the events with the lowest randomization factor.

The second step is the encoding of the selected cases as a transaction table, namely the data structure to be fed into EL-RM. The passage is necessary for third step to take place, as it is the association rule discovery by EL-RM. The fourth step is the post-processing step, wherein EC-SA-RM operates as follows. First, it ranks the discovered rules based on their confidence and lift [8]. Second, it filters out the rules with confidence equal to one to avoid that the subsequent iterations are prone to overfitting – such rules would naturally tend to restrict the exploration range closer to the previous individuals in the search space. Following the encoding and the mining steps, for example, we get the two rules depicted in Fig. 5(b). Third, it merges the filtered rules into new ones as the computation time for their checking is expensive and affected by the overall number of rules at hand. EC-SA-RM combines all the sets of rules that share the same antecedent (i.e., in the `IF` part) into new ones. For each of these new rules, the antecedent is the common one, and the consequent is the conjunction of the combined rules (i.e., in the `THEN` part). We formally define the rule that stems from this combination as follows.

Definition 5 (Combined rule): Let $\mathfrak{R} \ni R$ be a set of data rules expressed as in Eq. (1). Let $\sim_{\texttt{IF}} \in \mathfrak{R} \times \mathfrak{R}$ be the equivalence relation that includes in an equivalence class $[R]_{\texttt{IF}}$ the rules that have the same antecedent as R: $\sim_{\texttt{IF}} \triangleq \{(R, R') : R_{\texttt{IF}} \equiv R'_{\texttt{IF}}\}$. Given an equivalence class $[R]_{\texttt{IF}}$ *combined rule* $\tilde{R}$ is expressed as follows:

$$\tilde{R} \triangleq \texttt{IF}\ R_{\texttt{IF}}\ \texttt{THEN} \bigwedge_{R' \in [R]_{\texttt{IF}}} R'_{\texttt{THEN}} \tag{3}$$

Notice that the conjunction of all rules in an equivalence class $[R]_{\texttt{IF}}$ is logically equivalent to its combined rule $\tilde{R}$. Therefore, our approach replaces the set of discovered data rules with the set of combined rules stemming from each equivalence class. The two rules in Fig. 5(b), e.g., are combined into the one in Fig. 5(c).

C. Including New Data Rules for SA Iterations

After learning new data rules, EC-SA-RM includes them in the set coming from the previous iteration (or given as an initial input, if any, at the beginning). EC-SA-RM uses the rules to guide the generation of the new neighbor individual x' and improve the event correlation decision step by reducing the randomization factor over the events.

For example, the data rules in Fig. 5(c) are used to generate a new individual x'. As illustrated in Section IV-B, $P_{e_5} =$

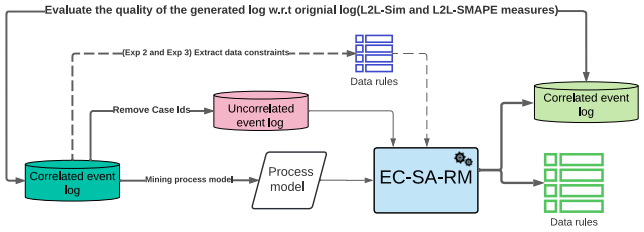


Figure 6: Evaluation steps

$\{\sigma_1, \sigma_2, \sigma_3\}$. The three cases are then ranked based on the number of satisfied data rules. Case σ_2 satisfies the combined rule, whereas σ_1 and σ_3 do not and are therefore ranked as the last. Thus, e_5 is assigned to σ_2 and it has a randomization factor of zero ($e_5.\mathrm{Rand} = 0$).

As shown in Fig. 4, EC-SA-RM proceeds until EC-SA-Data reaches the maximum number of iterations of the annealing process. Each iteration discovers new data rules using the previous iteration's selected individual and reassigns the events based on the given model and the new set of rules to explore the search space. We recall that accepting a worse solution than the previous one in some iterations is part of the rationale as it helps to skip the optimal local solution, with the aim to reach an approximate optimal global solution.

V. Evaluation

We implemented a prototype tool for EC-SA-RM.[1] Using this tool, we conducted three experiments to evaluate the accuracy of our approach, and compared the results with EC-SA [9] and EC-SA-Data [7] as a baseline.

A. Design

Figure 6 depicts our evaluation process. The primary input for the three experiments is a correlated event log. We refer to it as the *original log*. Using this log, we created an uncorrelated log by removing the case identifiers from it. Then, we mined the process models from the original logs using Split Miner [15]. We used four real-world datasets from the benchmark of Augusto et al. [16] based on the publicly available event logs in the BPIC repository. Table I shows some descriptive statistics about them.

We conducted three experiments, aimed at assessing the accuracy improvement our approach yields. The first experiment simulates the real-life scenario wherein the analyst has no prior knowledge about the data rules. We compare the results acquired with our approach with those of EC-SA, as the latter does not correlate the events based on the data rules knowledge. The second experiment mimics the common situation in which the analyst has *some* prior knowledge about the data rules. To this end, we extracted the data rules by visual inspection and analysis of those event logs. We compare the results acquired with our approach with those of EC-SA-Data, as the latter can use data rules to guide the correlation of the events. The third experiment performs a sensitivity analysis that investigates the effect of the initial set of data rules on the accuracy of our approach. We used the BPIC17 event log [17]. We examine the impact of increasing the number of input rules on the accuracy of the generated log. To this end, we gradually increment the number of used constraints from zero to ten and compare the results with those of EC-SA-Data.

[1] https://github.com/DinaBayomie/EC-SA-RM/releases/tag/EC-SA-RM-V1

Table I: Descriptive statistics of real logs

	Traces		Events		Trace length		
Event log	**Total**	**Dst.%**	**Total**	**Dst.%**	**Min**	**Avg**	**Max**
BPIC13$_{\mathrm{cp}}$ [18]	1487	12.3	6660	7	1	4	35
BPIC13$_{\mathrm{inc}}$ [19]	7554	20.0	65 533	13	1	9	123
BPIC15$_{1\mathrm{f}}$ [20]	902	32.7	21 656	70	5	24	50
BPIC17$_{\mathrm{f}}$ [17]	21 861	40.1	714 198	41	11	33	113

B. Accuracy metrics

To assess accuracy, we gauge the similarity between the original (correlated) log and the log generated by our technique using the four measures introduced in [7]. The first two measures focus on the structural similarity between generated and original logs. The other two measures take the temporal distance into account – namely, that of events' elapsed times, and cases' cycle times.

The first measure is the *bigram similarity* [7], which assesses the extent to which a generated log L' captures the event pair relationships in the original log L. It is based on the number of the pair of events (henceforth, bigrams, i.e., n-grams of length 2) that occur in both logs. We formally define it as follows.

Definition 6 (Bigram similarity): Let L and L' be two event logs. We denote with $\mathrm{occurs2}\left(\langle e, e'\rangle, L\right)$ the indicator function that returns 1 if there exists a case $\sigma \in L$ such that $\langle e, e'\rangle$ is a segment of it:

$$\mathrm{occurs2}(\langle e, e'\rangle, L) = \begin{cases} 1 & \text{if there exists } \sigma \in L \text{ s.t. } \langle e, e'\rangle \subseteq \sigma \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

The *bigram similarity* $\mathrm{L2L}_{\mathrm{2gram}}$ is computed dividing by the cardinality of L the average number of bigrams in the cases of L that also occur in L' as follows:

$$\mathrm{L2L}_{\mathrm{2gram}}(L, L') = \frac{1}{|L|} \sum_{\sigma \in L} \frac{1}{|\sigma|-1} \left(\sum_{i=1}^{|\sigma|-1} \mathrm{occurs2}\left(\langle \sigma(i), \sigma(i+1)\rangle, L'\right) \right) \quad (5)$$

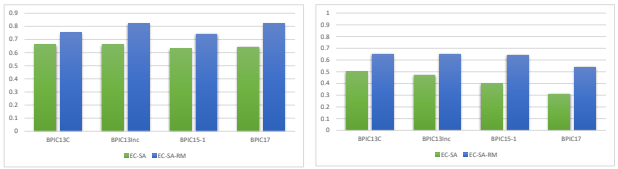
The second measure is the *case similarity* [7], which considers how many cases consist of identically correlated events in the original and correlated event logs.

Definition 7 (Case similarity, $\mathrm{L2L}_{\mathrm{case}}$*):* Given two event logs L and L', the *case similarity*, $\mathrm{L2L}_{\mathrm{case}}$, is the number of cases that are equal in L and L' divided by the total number of cases.

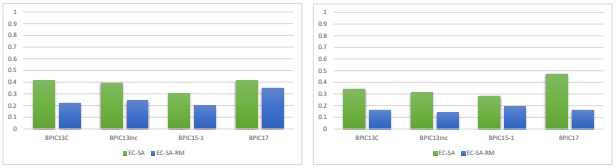
$$\mathrm{L2L}_{\mathrm{case}}(L, L') = \frac{|L \cap L'|}{|L|} \quad (6)$$

The third measure is the *event time deviation* [7], which considers in how far the generated log deviates in terms of the elapsed time of events from the original log. We formally define it as follows.

Definition 8 (Event time deviation): Let L and L' be event logs defined over a common universe of events E. Let $\mathrm{ET}(\sigma, e)$ be the elapsed time (ET), i.e., the execution duration of an

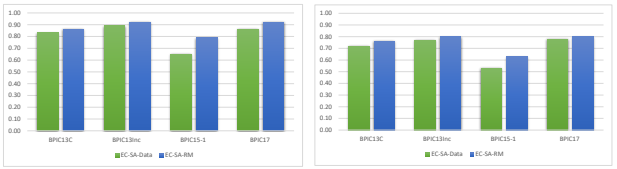


(a) $\text{L2L}_{\text{2gram}}$ (b) L2L_{case}



(c) SMAPE_{ET} (d) SMAPE_{CT}

Figure 7: Exp. 1: Impact of learning domain knowledge with no prior data rules



(a) $\text{L2L}_{\text{2gram}}$ (b) L2L_{case}

(c) SMAPE_{ET} (d) SMAPE_{CT}

Figure 8: Exp. 2: Impact of learning domain knowledge with prior data rules

event e in case σ, computed as follows:

$$\text{ET}(\sigma, e_{(\sigma,i)}) = \begin{cases} e_{(\sigma,i)}.\text{Ts} - e_{(\sigma,i-1)}.\text{Ts} & \text{if } 1 < i \leqslant n \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

The *event time deviation*, SMAPE_{ET}, is the Symmetric Mean Absolute Percentage Error (SMAPE) of the elapsed time of events between L and L':

$$\text{SMAPE}_{\text{ET}}(L, L') = \frac{\sum_{e \in E} \frac{|\text{ET}(\sigma,e) - \text{ET}(\sigma',e)|}{|\text{ET}(\sigma,e)| + |\text{ET}(\sigma',e)|}}{|E| - |L|} \quad \text{with } \sigma = L(e),\ \sigma' = L'(e) \tag{8}$$

The fourth measure is the *case cycle time deviation* [7], which investigates the deviation of the generated log from the original one in terms of the cases' cycle time. To compare pairs of cases, we consider those that have the same start event. We formally define the measure as follows.

Definition 9 (Case cycle time deviation): Let $\text{CT}(\sigma)$ be the cycle time of a case σ, computed as follows [21], [22]:

$$\text{CT}(\sigma) = \sigma(|\sigma|).Ts - \sigma(1).Ts \tag{9}$$

Given two event logs L and L', the *case cycle time deviation* SMAPE_{CT} is the symmetric mean absolute percentage error of the cycle time between cases in L and L':

$$\text{SMAPE}_{\text{CT}}(L, L') = \frac{1}{|L|} \times \sum_{\substack{\sigma \in L, \\ \sigma' in L': \\ \sigma(1) = \sigma'(1)}} \frac{|\text{CT}(\sigma) - \text{CT}(\sigma')|}{|\text{CT}(\sigma)| + |\text{CT}(\sigma')|} \tag{10}$$

Notice that SMAPE_{ET} and SMAPE_{CT} are error measures, so low values reflect a higher quality of the results.

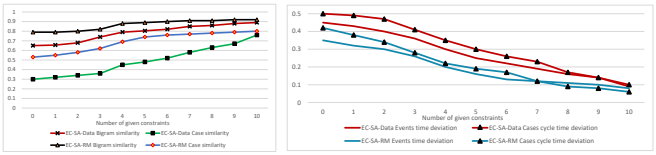
C. Results

Figure 7 depicts the results of the first experiment. Its aim is analyzing the impact of learning new data rules (with no prior ones as input) on the correlation accuracy. To this end, results are compared with the EC-SA, which, in contrast, does not consider any domain knowledge in addition to the process model for the correlation decision. We can see that learning the data rules and using them for the event correlation decision improves the accuracy. Indeed, EC-SA-RM outperforms EC-SA. Figures 7(a) and 7(b) show that $\text{L2L}_{\text{2gram}}$ and L2L_{case} exhibit an increment of around $14\,\%$ and $20\,\%$ on average, respectively. Notably, discovering the data rules over the iterations in EC-SA-RM dramatically improves the correlation quality over the BPIC17 and BPIC15 logs as it can be observed in Fig. 7(b) – notice that L2L_{case} increase by $24\,\%$ and $23\,\%$, respectively. Figures 7(c) and 7(d) highlight that also the time deviation decreases when rules are in use, as SMAPE_{ET} and SMAPE_{CT} go down by $13\,\%$ and $19\,\%$.

Notice that learning and using new data rules over subsequent SA iterations enhances the correlation process by decreasing the randomization factor of event-case assignments, as it prunes the violated candidate cases. Furthermore, EC-SA-RM builds the data rules that describe the generated event log and justify the event assignment decision. For instance, one of the rules returned by EC-SA-RM for BPIC17 explains the correlation in $14\,\%$ of the events: IF $\sigma(i).\text{Act} =$ 'O_Sent (mail and online)' $\wedge\ \sigma(j).\text{Act} =$ 'O_Created' THEN $\sigma(i).\text{OfferID} = \sigma(j).\text{OfferID} \wedge \sigma(i).\text{Type} = \sigma(j).\text{Type}$.

Figure 8 illustrates the results of the second experiment, which studies the impact of learning new data rules with prior data rules given as input, and compares the results with EC-SA-Data. We can see that learning new data rules over the iterations improves the accuracy. Figures 8(a) and 8(b) show that $\text{L2L}_{\text{2gram}}$ and L2L_{case} increase by around $7\,\%$ and $5\,\%$ on average, respectively. Notably, discovering the data rules over the iterations in EC-SA-RM dramatically improves the correlation quality over the BPIC15 log, as L2L_{case} improves by $10\,\%$ as depicted in Fig. 8(b). Figures 8(c) and 8(d) put in evidence that also the time deviation decreases, as SMAPE_{ET} and SMAPE_{CT} decrease by $5\,\%$ and $4\,\%$, respectively.

The usage of rule mining in EC-SA-RM affects execution time performance. The reason is, every iteration awaits for the discovery of the new data rules. As a result, the overall computation time is higher than EC-SA-Data. For instance, EC-SA-RM ran for $20\,\text{h}$ to complete the execution with the BPIC17, whereas EC-SA-Data took $13.7\,\text{h}$ to complete. The processing of the $\text{BPIC15}_{1\text{f}}$ log required $12\,\text{h}$, whilst EC-SA-Data needed $6.5\,\text{h}$. We remark that the final set of data rules learned by EC-SA-RM is expected to be different from the final set of rules learned over the original log by



(a) $\mathrm{L2L}_{\mathrm{2gram}}$ and $\mathrm{L2L}_{\mathrm{case}}$ (b) $\mathrm{SMAPE}_{\mathrm{ET}}$ and $\mathrm{SMAPE}_{\mathrm{CT}}$

Figure 9: Exp. 3: Sensitivity analysis on the impact of prior data rules on accuracy

EL-RM, given that we still cannot reach a case similarity ($\mathrm{L2L}_{\mathrm{case}}$) of 100 %.

Figure 9 depicts the results of the third experiment, which studies the effect of the number of input data rules on the accuracy of the correlation process, and compares the results with EC-SA-Data. We can see that having an initial set of data rules is beneficial to the discovery phase as they offer a better guidance. Indeed, the technique convergences faster towards the result and the accuracy of the generated log increases. Figure 9(a) shows that $\mathrm{L2L}_{\mathrm{2gram}}$ and $\mathrm{L2L}_{\mathrm{case}}$ increase by around 13 % and 27 % on average, respectively. Furthermore, EC-SA-RM outperfoms EC-SA-Data by around 8 % and 20 % on average, respectively. Figure 9(b) highlights that also the time deviation decreases when constraints are in use, as $\mathrm{SMAPE}_{\mathrm{ET}}$ and $\mathrm{SMAPE}_{\mathrm{CT}}$ decrease by around 27 % and 36 % on average, respectively. Also, EC-SA-RM outperfoms EC-SA-Data by around 8 % and 10 % on average, respectively.

D. Discussion

Our experiments show that learning and using data rules improves correlation accuracy due to a reduction of the randomization factor for event-case assignment. Our approach is sensitive to the accuracy of the given input, as the process model and the initial set of data rules influence the event correlation decision. However, discovering new rules over every iteration may balance the negative impact of an inaccurate input. EC-SA-RM, thus, is flexible concerning the prior knowledge of the analyst. The learned data rules let EC-SA-RM improve the accuracy of the generated log, as evidenced by all experiments (the first one having no rules provided as input, and the other two resorting to partial prior knowledge). Last but not least, they provide an explanation for the correlation decisions, which can support the process analyst for further analyses.

VI. Conclusion

We presented EC-SA-RM, a technique for the automated correlation of events. Our approach learns data rules to integrate domain knowledge discovered at run time with the given input and thereby drive the correlation decisions. A key quality that EC-SA-RM enjoys is thus its flexibility concerning the prior knowledge of the analyst, on which other techniques heavily rely instead. Our findings show that EC-SA-RM is able to learn a more accurate event log when compared with state-of-the-art algorithms. In addition, the returned data rules can be used as a means to illustrate the rationale behind the assignment of cases to events, thereby equipping our technique with an additional explainability lens.

As future work, we want to include measures dedicated to the direct evaluation of the impact of the rules over the iterations. Also, we plan to use these measures to justify the correlation decisions and therefore provide more accurate explanation for the process analysts. Moreover, we aim to deepen the combined data analysis from multiple process perspectives and modeling paradigms at once [23], [24].

References

[1] W. van der Aalst, *Process Mining – Data science in action*. Springer, 2016.

[2] D. R. Ferreira and D. Gillblad, "Discovering process models from unlabelled event logs," in *BPM*, 2009, pp. 143–158.

[3] S. Pourmirza, R. Dijkman, and P. Grefen, "Correlation Miner: Mining business process models and event correlations without case identifiers," *IJCIS*, no. 02, 2017.

[4] A. Abbad Andaloussi, A. Burattin, and B. Weber, "Toward an Automated Labeling of Event Log Attributes," in *BPMDS*, 2018, pp. 82–96.

[5] S. Bala, J. Mendling, M. Schimak, and P. Queteschiner, "Case and activity identification for mining process models from middleware," in *PoEM*, 2018, pp. 86–102.

[6] D. Bayomie, A. Awad, and E. Ezat, "Correlating unlabeled events from cyclic business processes execution," in *CAiSE*, 2016, pp. 274–289.

[7] D. Bayomie, C. Di Ciccio, and J. Mendling, "Event-case correlation for process mining using probabilistic optimization," 2022. [Online]. Available: https://arxiv.org/abs/2206.10009

[8] D. Bayomie, K. Revoredo, and J. Mendling, "Multi-perspective process analysis: Mining the association between control flow and data objects," in *CAiSE*, 2022, pp. 72–89.

[9] D. Bayomie, C. D. Ciccio, M. La Rosa, and J. Mendling, "A probabilistic approach to event-case correlation for process mining," in *ER*, 2019, pp. 136–152.

[10] H. Nezhad, R. Saint-Paul, F. Casati, and B. Benatallah, "Event correlation for process discovery from web service interaction logs," *VLDB J.*, no. 3, 2011.

[11] E. G. L. de Murillas, H. A. Reijers, and W. M. van der Aalst, "Case notion discovery and recommendation: automated event log building on databases," *Knowl. Inf. Syst.*, 2019.

[12] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques, 3rd edition*. Morgan Kaufmann, 2011.

[13] L. Geng and H. J. Hamilton, "Interestingness measures for data mining: A survey," *ACM Comput. Surv.*, vol. 38, no. 3, p. 9, 2006.

[14] A. Askarzadeh, L. dos Santos Coelho, C. E. Klein, and V. C. Mariani, "A population-based simulated annealing algorithm for global optimization," in *SMC*, 2016, pp. 4626–4633.

[15] A. Augusto, R. Conforti, M. Dumas, M. La Rosa, and A. Polyvyanyy, "Split Miner: Automated discovery of accurate and simple business process models from event logs," *Knowl. Inf. Syst.*, 2018.

[16] A. Augusto, R. Conforti, M. Dumas, M. La Rosa, F. Maggi, A. Marrella, M. Mecella, and A. Soo, "Automated discovery of process models from event logs: Review and benchmark," *IEEE TKDE*, no. 4, 2019.

[17] "BPI Challenge 2017 – Offer log," 2021. [Online]. Available: https://doi.org/10.4121/12705737.v2

[18] "BPI Challenge 2013, closed problems," 2013. [Online]. Available: https://doi.org/10.4121/uuid:c2c3b154-ab26-4b31-a0e8-8f2350ddac11

[19] "BPI Challenge 2013, incidents," 2013. [Online]. Available: https://doi.org/10.4121/uuid:500573e6-accc-4b0c-9576-aa5468b10cee

[20] "BPI Challenge 2015 – Municipality 1," 2015. [Online]. Available: https://doi.org/10.4121/uuid:a0addfda-2044-4541-a450-fdcc9fe16d17

[21] A. Van Looy and A. Shafagatova, "Business process performance measurement: a structured literature review of indicators, measures and metrics," *SpringerPlus*, no. 1, p. 1797, 2016.

[22] M. Dumas, M. La Rosa, J. Mendling, and H. A. Reijers, *Fundamentals of Business Process Management, Second Edition*. Springer, 2018.

[23] F. Mannhardt, M. de Leoni, H. A. Reijers, and W. M. P. van der Aalst, "Balanced multi-perspective checking of process conformance," *Computing*, vol. 98, no. 4, pp. 407–437, 2016.

[24] B. F. van Dongen, J. De Smedt, C. Di Ciccio, and J. Mendling, "Conformance checking of mixed-paradigm process models," *Inf. Syst.*, vol. 102, p. 101685, 2021.