

# Use of FURIA for Improving Task Mining

Petr Průcha , Jan Skrbek 

Faculty of Economics, Technical University of Liberec, Studentská 1402/2, 461 17 Liberec, Czech Republic

Corresponding author: Petr Průcha ([petr.prucha@tul.cz](mailto:petr.prucha@tul.cz))

---

## Abstract

Companies that use robotic process automation very often deal with the problem of selecting a suitable process for automation. Manual selection of a suitable process is very time-consuming. Therefore, part of the process mining field specializes in selecting suitable processes for automation based on process data. This work deals with the possibility of improving the existing method for finding suitable candidates for automation. To improve the current approach, we remove the limiting restrictions of the current method and use another FURIA rule-learning algorithm for rule detection. We use three different datasets and the WEKA platform to validate the results. The results show that FURIA and the removal of strictly deterministic rules as restrictions turned out to be a competitive approach to the original one. On data presented in this study, the selected approach detected more candidates for automation and with higher accuracy. This study implies that FURIA and not using a strictly deterministic process is an appropriate procedure with certain use cases as other procedures mentioned in this study.

## Keywords

FURIA; Task mining; RPA; Robotic process automation; RIPPER; Automatable routines.

---

# 1 Introduction

Today, many organizations are trying to minimize costs and eliminate the increasing number of administrative tasks. Some tasks need to be performed, for example, for legislative reasons or because they are essential for the organization's operation. Routine and administrative tasks can be automated using current technologies, such as connecting applications via APIs or robotic process automation (RPA) technology. RPA robots are able to perform routine activities just like a computer user. RPA technology has been gaining a lot of attention lately. However, RPA technology also has its limits, and one of the problems is selecting a suitable activity or process to automate so-called task mining (Syed et al., 2020).

Task mining is a sub-area of process mining that focuses on finding suitable processes and tasks for automation. A comparison of task-mining approaches is presented in Table 1, where the procedures used, the data used and the authors of the work are described. Most authors use UI logs to select candidates for automation.

This research is based on the work of Bosco et al. (2019) and subsequently builds on his work. In this research, another approach will be introduced and tested that may bring better results and thus improve the current algorithm of Bosco et al. (2019).

Bosco et al. (2019) seek to discover deterministic processes for automation by comparing UI logs with previous logs and also by using machine learning algorithms such as RIPPER by Cohen (1995) and FOOFAH by Jin & Anderson (2017). Working with deterministic processes, which are 100% able to determine whether activities are automated, significantly narrows potential candidates for automation. In the research, we adjust this assumption in connection with other scientists' knowledge and practical knowledge. By not using strictly deterministic rules, we will expand the circle of potential candidates for automation. We also test the accuracy of the rules obtained by the FURIA rule-learning algorithm (RLA) compared to the rules obtained from the RLA RIPPER. FURIA is a newer algorithm than RIPPER. Both these algorithms fall into the category of RLA algorithms (Hühn & Hüllermeier, 2010). FURIA uses fuzzy logic to search for rules in the data, which is why its use appears to be a better RLA than RIPPER.

This article aims to test the possibility of using not strictly deterministic rules to identify candidates for automation and to compare the FURIA algorithm with RIPPER.

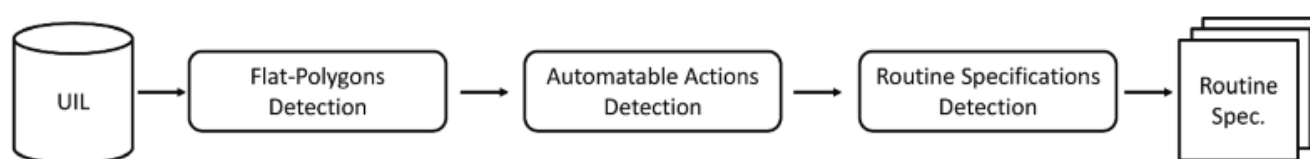
**Table 1.** Comparison of task mining approaches.

Approach	Type of data/example where the principle is used	Are the selected candidates strictly deterministic?	Authors
Use strictly deterministic rules. Use FOOFAH algorithm to discover rules (Jin & Anderson, 2017).	UI logs – Try to find rules for transforming data between Excel tables.	Yes	(Leno et al., 2021a)
Try to find formula in data with most similarity and based on that. Generate routines for automation. Use PM4PY and a-priori.	UI logs – Move data from Excel spreadsheet to web form.	No	(Agostinelli et al., 2020)
Present a method of selecting suitable candidates for automation based on process visualization and presented patterns.	UI logs – Download data from ERP to Excel and perform data transformation in Excel.	No	(Choi et al., 2021)

Approach	Type of data/example where the principle is used	Are the selected candidates strictly deterministic?	Authors
Present methods for evaluating candidates for automation based on selected factors, which are evaluated using marks. Resulting marks serve as suitability evaluation factor.	Process logs (not exactly specified) – Tested on nine different examples.	No	(Viehhauser & Doerr, 2021)
Discover routines and sequences that are strictly deterministic, use RIPPER, FOFAH algorithms.	UI logs – Rewriting input data from study department into web form and Excel.	Yes	(Bosco et al., 2019)
Select and discover routines based on the CloFast algorithm (Fumarola et al., 2016). Use the algorithm to find similar sequences that meet a certain threshold. The sequence is then evaluated based on specific criteria.	UI logs – example based on data from web form and Excel spreadsheet.	Yes	(Leno et al., 2021b)
Using AI to process so-called NPL based on text inputs predicts whether activity is automatable.	Process logs– 47 different processes from 10 different sources	No	(Leopold et al., 2018)

## 1.1 Bosco's approach

As mentioned in the Introduction, this research extends the work of Bosco et al. (2019) and describes the approach they use in more detail. Their approach is described in Figure 1. In the first phase, they sorted UI logs from the process. They then detected flat-polygons from the UI logs using the Deterministic Acyclic Finite State Automaton (DAFSA) method – which maps all process paths that are in the process. The output from the flat-polygon detection can be seen in Figure 2. After finding all process paths, it is detected whether the given action is automatable, including all the parameters of the given action. Each action has multiple parameters (at least one) depending on the user activity and the application used. From the authors' point of view, the action is automatable if all parameters can be determined in each of its cases, in which case we can consider the action deterministic. They use three approaches to find out the parameters. The first approach is based on repeating the parameters of the action during the process. This means that the action parameter is always the same in all cases. For more complex cases that cannot be determined by the first procedure, the authors use two algorithms. The one to find rules in the data is called RIPPER. The second algorithm is called FOFAH and is used to find rules for data transformation. It is used mainly in connection with spreadsheets.



**Figure 1.** Bosco's approach to discover automatable routines. Source: Bosco et al. (2019).

The fourth part of the procedure is to add information to the actions and obtain additional information to specify the candidate's routine. In the fourth part of the algorithm, the actions are combined into consecutive routines. Rules are then searched for these routines. They use the RIPPER algorithm to search for rules. If RIPPER does not find the rule, it creates a trivial condition by shortening the routine by the first action, which will then be the trigger in this case.

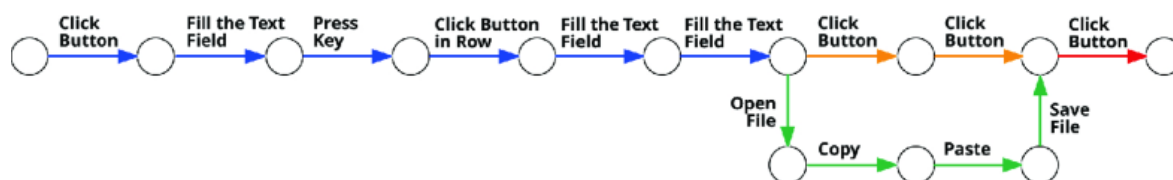


Figure 2. Working example in Bosco's article. Source: Bosco et al. (2019).

### 1.1.1 Bosco's working example

An example in which Bosco et al. (2019) test their approach can be seen in Figure 2, which is based on the process of the Department of Studies at the University of Melbourne, where study department staff update student information. The first part (blue) is always the same: the student comes to the study department, the student confirms his information, and the staff feeds the information to the system. Subsequently, if the student is Australian, he/she will confirm his/her address, which the study department backs up in Excel (green variant). If he/she is of a different nationality, the staff will check the foreign student option in the information system (orange variant). The last part (red) of the process is just finishing the completion by clicking a button.

## 1.2 Approach

As indicated in the Introduction, the approach of Bosco et al. (2019) is based on strictly deterministic rules. According to its interpretation, it is possible to automate a given activity only in the case of a strictly deterministic process. Candidates in Bosco et al. (2019) are selected based on these criteria. However, this approach severely limits the selection of potential candidates for automation, especially if the rules have to be discovered by an algorithm, because in many cases, as shown by interviews with RPA experts and even scientific articles in this field, RPA robots often work with applications that act as a "black box". This means that application users do not know the rules by which the application generates results. This is typical of legacy systems, which are very often automated using RPA. Another reason why we will not use only strictly deterministic rules in our approach is because quite often, only certain parts of the process are automated, and the rest of the process is taken care of by a person. In the RPA industry, processes that are not automated throughout the process are commonly used, and so-called attended RPA robots are used, where a human assists the RPA robot. There are several scenarios and ways in which this collaboration takes place: for example, a person performs a part of a process, and then the robot completes the activities; another possibility is that the robot cannot process, mark and continue, or the robot falls into an exception if it cannot meet the case. A person will then complete notable cases and cases that ended with an exception (Leno et al., 2021; Soeny et al., 2021; Syed et al., 2020).

Using the above example from Bosco et al. (2019), it is possible to show how limiting the use of strictly deterministic processes is. An example can be shown in parameter 30, which is in Table 2. Parameter 30 has the values Australia or country with a random number. The rule discovered by the RIPPER algorithm is rewritten into pseudocode for a better understanding, the original rule can be found in Appendix A.

```

If Parameter 8 == ID1103
    Parameter 30 = Country5
Else:
    Parameter 30 = Australia

```

The RIPPER algorithm was able to use this rule to determine approximately 50% of cases correctly. It can be assumed from that that 50% of all cases can be automated, although in this case, the rule found by RIPPER is not appropriate as this 50% determination is very random. The FURIA algorithm generated better rules. The original rule can be found in Appendix A, as the rule has been rewritten into pseudocode. The accuracy of the FURIA algorithm rules is 50%.

```

If parameter36 == C:/Customers/Australia/
    Parameter30 = Australia
If Parameter8 == ID1103
    Parameter30 = Country5
If Parameter8 == ID396
    Parameter30 = Country140

```

These examples show that strictly deterministic rules that satisfy the condition of `confidence == 1.0` from Bosco et al. (2019) are restrictive. This is mainly why some approaches in Table 1 do not use strictly deterministic rules to select suitable candidates. For the approach used by Bosco et al. (2019), the circle of candidates could simply be extended by changing the condition to, for example, `confidence == 0.5`. The value of confidence will vary depending on the industry, the number of cases, the overall profitability and other benefits that automation can bring (Aguirre & Rodriguez, 2017; Syed et al., 2020; van der Aalst et al., 2018).

In our approach, we will expand the circle of potential candidates for automation with candidates that are not strictly deterministic, as we assume that at least a part of the process can be automated, even if we do not know the rules that would satisfy 100% of cases.

## 2 Methodology

This research, as mentioned above, follows the procedures of Bosco et al. (2019), where different methods are used to determine whether the current approach can be improved. In this research, we focus on only a part of the process from Bosco et al. (2019), specifically on the key part of the whole approach, namely the detection of automation actions described in 1.1 above. This part is crucial due to the fact that many actions are determined to be non-automated due to failure to find rules for 100% determination of the parameters of the action. For this research, we will remove this limitation, because it follows from the literature search and communication with RPA specialists that many processes are only partially automated and only for some cases. In this research, we work on the assumption that actions whose parameters cannot be 100% determined will continue to be considered automated because RIPPER or another RLA algorithm can correctly determine some parameters, for example, in 50% of cases.

This research will test the RLA FURIA algorithm on the example and data mentioned in 1.1 above. The research aims to compare whether RLA FURIA is better than RLA RIPPER.

### Research question:

*Is the FURIA algorithm better than RIPPER when searching for rules in UI logs?*

To answer this question, we will use the UI logs `log6.mxml` and `log6.mxml` from Bosco et al. (2019) and the UI logs presented in 2.1 below. `Log6.mxml` was chosen because it represents Bosco's real example of the process. `Log9.mxml` was selected because of the highest complexity and highest number of records.

For comparison of FURIA and RIPPER, we used the WEKA platform. The WEKA (Waikato Environment for Knowledge Analysis) platform version 3.8.5, installed on a Windows 10 computer, was used to calculate and search for rules. The WEKA platform integrates the RLA algorithms RIPPER (jRip) and FURIA. The basic settings in the WEKA program are used to determine the rules.

The data<sup>1</sup> in log6.mxlm were converted to CSV format using a custom program. The data and the program can be found in the repository under the name log6.csv and main.py. An example of transformed data named log6.csv is in Table 2 as transformed data. The second line of Table 2 is not included in the testing dataset. The row helps understand the transformation. The original MXML data format is the XML version. A variation of the XML format called XES is widely used in the process mining field. The XES format is a process mining standard based on IEEE standards. XES is suitable for transmitting event logs, because they faithfully capture reality. XES also supports most process mining tools (Narayana et al., 2020). Although MXML and XES are XML formats, the XML format is often not suitable for further analysis and therefore, process-mining tools offer XML and XES to CSV transformations. The conversion of MXML to CSV was performed as follows.

The MXML log is made up of an `AuditTrailEntry`, which is then made up of the `<Data>` tag and other components behind the `</Data>` tag. For our research, we were mainly interested in the values within the `<Data>` and `</Data>` tags. We used the attribute name value in the abstract as the column name (second row of Table 2) and the value inside the attribute tag as the value for that column. Each additional action data item in the `AuditTrailEntry` for the related process record is sorted in the following column. Each line indicates one process record, which consists of multiple actions. Throughout the research, this format/approach is used. Each column is marked as a parameter with the number of that column. The attribute name values are not used because they do not carry key information, and essential data are used for our research. For example, in the DISCO process-mining tool, automated XES to CSV conversion uses the attribute name value and the tag value. It joins them together using the ":" character. So the values of the first three values would look like this:

Label:value1, Value:value336, source:Web, ...

MXML format example:

```
<AuditTrailEntry>
  <Data>
    <attribute name="Label">value1</attribute>
    <attribute name="Value">value336</attribute>
    <attribute name="source">Web</attribute>
    <attribute name="lifecycle:transition">complete</attribute>
    <attribute name="concept:name">insertValue</attribute>
    <attribute name="time:timestamp">2019-03-05T11:42:31.306+11:00</attribute>
  </Data>
  <WorkflowModelElement>insertValue</WorkflowModelElement>
  <EventType>complete</EventType>
  <timestamp>2019-03-05T11:42:31.306+11:00</timestamp>
</AuditTrailEntry>
```

Parameter 36 will be tested for log6.csv data, as this parameter is the only rule that was discovered by the original algorithm. In addition, parameters that do not contain a timestamp or do not have identical values in the entire column will be tested. The last criterion is that the column must contain values in all rows, because for these results, these remaining parameters would be uniquely determined to be 100% except for the value of the timestamp and parameter 51. All the tested parameters are in Table 4.

<sup>1</sup> The data can be downloaded from: <https://github.com/Scherifow/SGS-Task-Mining>

**Table 2.** Sample data – log6.csv. Source: data transformed into CSV by Bosco et al. (2019).

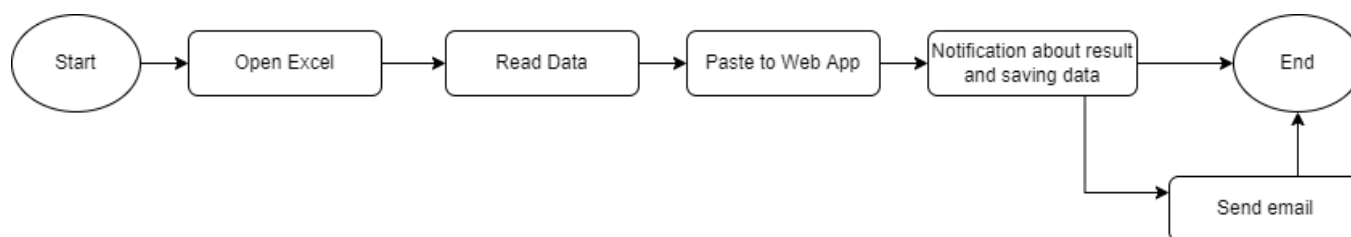
Parameter 28	Parameter 29	Parameter 30	Parameter 31	Parameter 36
Action type	Timestamp	Value	Label	Path
insertValue	2019-03-05T11:48:53.780+11:00	Country169	country	UnbindFile
insertValue	2019-03-05T11:49:58.732+11:00	Country183	country	UnbindFile
insertValue	2019-03-05T11:50:51.115+11:00	Australia	country	C:/Customers/Australia/
insertValue	2019-03-05T11:51:53.245+11:00	Country164	country	UnbindFile
insertValue	2019-03-05T11:52:53.824+11:00	Country190	country	UnbindFile
insertValue	2019-03-05T11:53:28.975+11:00	Australia	country	C:/Customers/Australia/
insertValue	2019-03-05T11:54:34.283+11:00	Country87	country	UnbindFile
insertValue	2019-03-05T11:55:17.999+11:00	Australia	country	C:/Customers/Australia/
insertValue	2019-03-05T11:56:42.956+11:00	Country68	country	UnbindFile

For the dataset described in 2.1 below, the data are already in CSV format, so no further transformation is required. For these data, parameter 37 has the most significant value because the remaining parameters are identical, or they are input data; see parameter 7 to parameter 31.

## 2.1 Working example

Furthermore, data were created for the research based on a real process, which was already automated using RPA; however, in the past, an employee performed this activity. Therefore, it is no longer possible to obtain real UI logs on which to conduct research. The data<sup>2</sup> were created from a dataset by Bosco et al. (2019) and the 1985 Auto Imports Database by Schlimmer. It was a process where an employee opened an Excel form, loaded and copied the data into a web application, which processed the data (as a black box), displayed the result and saved the data, and according to the result, the web application sent an email. Then it was prepared for entering new data or ended in the previous step. See Figure 3 for the process visualization.

<sup>2</sup> The data can be found in the repository: <https://github.com/Scherifow/SGS-Task-Mining>



**Figure 3.** Working example of the process for our data.

### 3 Results

The results obtained when testing parameter 36 on the data named log6.csv can be seen in Table 3. The accuracy of the FURIA and RIPPER algorithms is 100%, so for this type of data, FURIA has the same results as RIPPER. The number of records or instances is 999 in the data, and both algorithms determined all instances correctly, thanks to the rules below.

**Table 3.** Algorithm results on log6.csv data.

	FURIA	RIPPER
Accuracy	100%	100%
Correctly classified instances	999	999
Incorrectly classified instances	0	0

The rules discovered in the log6.csv data by the RIPPER algorithm were only two, and they were also discovered by the default program. These rules can be written as follows:

```

If Parameter 30 == Australia:
    Parameter 36 = C:/Customers/Australia/
If Parameter 30 != Australia:
    Parameter 36 = UnbindFile
  
```

FURIA discovered slightly different rules in the same data with the same result, which means that the algorithms for this type of data are identically accurate.

```

If Parameter 37 == Web:
    Parameter 36 = UnbindFile
If Parameter 30 == Australia:
    Parameter 36 = C:/Customers/Australia/
  
```

Table 4 shows the accuracy results for all the rules that the algorithms were able to find for the given parameter. All the parameters were selected by rules stated in Section 2. In many cases, the accuracy of the algorithms was the same except for the parameters stated in Table 4. The dataset log6.csv contains only one parameter that has different results. The dataset log9.csv contains 17 parameters with different results. The most significant differences are in parameter64 and parameter61. FURIA is more accurate in 17 out of 18 cases displayed in the Table 4. RIPPER was more accurate only in one case parameter36.



**Table 4.** Results of comparison of RIPPER and FURIA in log6.csv and log9.csv data.

Algorithm	Dataset	Parameter	Correctly classified instances	Incorrectly classified instances	Correctly classified instances in %	Incorrectly classified instances in %
FURIA	log6.csv	42	513	486	51.35	48.65
RIPPER	log6.csv	42	506	493	50.65	49.35
FURIA	log9.csv	65	1057	534	66.44	33.56
RIPPER	log9.csv	65	1056	535	66.37	33.62
FURIA	log9.csv	64	1270	729	53.53	36.47
RIPPER	log9.csv	64	1128	871	56.43	45.57
FURIA	log9.csv	61	1503	496	75.19	24.81
RIPPER	log9.csv	61	1467	532	73.39	26.61
FURIA	log9.csv	59	71	1928	3.55	96.45
RIPPER	log9.csv	59	65	1934	3.25	96.75
FURIA	log9.csv	57	939	1060	46.97	53.03
RIPPER	log9.csv	57	938	1061	46.92	53.08
FURIA	log9.csv	55	635	1364	31.77	68.23
RIPPER	log9.csv	55	620	1379	31.02	68.98
FURIA	log9.csv	53	1104	895	55.23	44.77
RIPPER	log9.csv	53	1102	897	55.13	44.87
FURIA	log9.csv	51	577	1422	28.86	71.14
RIPPER	log9.csv	51	575	1424	28.76	71.24
FURIA	log9.csv	49	1130	869	56.53	43.47
RIPPER	log9.csv	49	1100	899	55.03	44.97
FURIA	log9.csv	48	1967	32	98.4	1.6
RIPPER	log9.csv	48	1961	38	98.1	1.9
FURIA	log9.csv	45	1103	896	55.18	44.82
RIPPER	log9.csv	45	1095	904	54.78	45.22
FURIA	log9.csv	43	1929	70	96.5	3.5
RIPPER	log9.csv	43	1920	79	96.05	3.95
FURIA	log9.csv	42	1512	487	75.64	24.36
RIPPER	log9.csv	42	1506	493	75.34	24.66
FURIA	log9.csv	37	1423	576	71.19	28.81
RIPPER	log9.csv	37	1414	585	70.74	29.26
FURIA	log9.csv	36	1737	262	86.89	13.11
RIPPER	log9.csv	36	1748	251	87.44	12.56
FURIA	log9.csv	9	182	1817	9.1	90.9
RIPPER	log9.csv	9	181	1818	9.05	90.95
FURIA	log9.csv	3	182	1817	9.1	90.9
RIPPER	log9.csv	3	181	1818	9.05	90.95

The following results were found in the data from Section 2.2 titled Auto2Mail.csv; see Table 5.

**Table 5.** Results of sample data Auto2Mail.csv

Algorithm	Correctly classified instances	Incorrectly classified instances	Correctly classified instances in %	Incorrectly classified instances in %
FURIA	182	23	88.78	11.22
RIPPER	164	41	80	20

On these data, the rules of the FURIA algorithm are clearly more accurate than RIPPER. The rules for these cases can be found in Appendix A. FURIA correctly identified 182 cases out of 205 cases, and the accuracy is 88.8%. RIPPER correctly identified 164 cases out of 205, so the accuracy is 80%. The FURIA algorithm is more accurate than RIPPER on these data. In Table 5, the results are rounded to two decimal places.

## 4 Discussion

Task mining is a hot topic, and the moment someone introduces a universal algorithm that can find processes to automate in existing data, this technology will be widely used across organizations that will be willing to pay a lot of money for it. Since this is a solution to a lucrative problem, researchers in the commercial sphere are also trying to find a universal algorithm for finding routines for automation. Start-ups and especially big players in RPA automation or process mining as UiPath or Celonis deal with this issue. Unfortunately, their task-mining procedures are inaccessible to the scientific community, and they are trade secrets.

One of the problems with task mining is data and their quality (Leno et al., 2020). From a data point of view, it is problematic that the methods mentioned above use the user's UI logs. The main problem with UI logs is that they are not monitored by default in most companies, such as process logs from ERP systems. The problem with not monitoring by default is that it leads to suspicious behaviour of employees whose computer activities are monitored (Razaghpanah et al., 2018).

With poor communication from the organization, this leads to the employee being intentionally sabotaging logging. The data quality is then inferior, and the preparation of data for analysis is difficult, sometimes impossible. However, data quality is problematic even without intentional sabotage, as people are often disturbed at work and perform the same tasks in a different order with the same result. Furthermore, even a simple decision made by a person can be very difficult to detect by artificial intelligence (Leno et al., 2021b). Another critical factor is that UI logs have more parameters than classic process logs, which are, in many cases, very strict.

An interesting approach to detecting automatable routines is to use textual descriptions of the process that can be processed and use them to identify the complexity of the process and whether it is suitable for automation (Leopold et al., 2018). Their estimates of whether the process is automatable are based on text inputs/data and an AI algorithm.

It is also important for our approach to mention that the results of RLA algorithms may differ based on input data, and each algorithm is differently accurate for different data, and it cannot be said that FURIA is better in all cases for all types of data. (Hühn & Hüllermeier, 2009; Manghai & Jegadeeshwaran, 2019). Therefore, the results of our research have certain limitations associated with the dataset used. It is thus appropriate for future research to test this approach on other data that will provide a more comprehensive view of this approach.

## 5 Conclusion

Research has shown that strictly deterministic rules are restrictive and limit the number of candidates for automation. A real example and the approaches of other experts confirm that it is possible to automate processes only for certain instances, that no algorithms need to discover the rules of the process, and that it is appropriate to expand the number of possible candidates for automation. Furthermore, the FURIA RLA algorithm has been shown to be much more accurate than RIPPER on tested data. However, as other experts mention, the data are fundamental. Furthermore, FURIA was able to find rules on test data that are more suitable for RPA automation. Thus, this research shows that the extension of potential candidates by not being strictly deterministic and using the FURIA algorithm is beneficial.

## Additional Information and Declarations

**Acknowledgements:** Special thanks belongs to Tereza Semerádová and Tereza Vrbová.

**Funding:** This work was funded by the SGS, Grant No. 2021-1033.

**Conflict of Interests:** The authors declare no conflict of interest.

**Data Availability:** The data that support the findings of this study are available from: <https://doi.org/10.6084/m9.figshare.7850918.v1>. Transformed data and algorithms from (Bosco et al., 2019) for this research are available from: <https://github.com/Scherifow/SGS-Task-Mining>.

## Appendix A

### JRIP rules:

```
(Parameter8 = ID1103) => Parameter30=Country5 (2.0/0.0)
=> Parameter30=Australia (997.0/499.0)
```

### FURIA rules:

```
(Parameter36 = C:/Customers/Australia/) => Parameter30=Australia (CF = 1.0)
(Parameter8 = ID1103) => Parameter30=Country5 (CF = 0.5)
(Parameter8 = ID396) => Parameter30=Country140 (CF = 0.4)
```

## Appendix B

### FURIA rules:

```
(Parameter11 = two) and (Parameter1 in [70, 73, inf, inf]) => Parameter37=Sent email
(CF = 0.98)
(Parameter16 in [-inf, -inf, 1732, 1867]) and (Parameter15 in [-inf, -inf, 953,
963]) and (Parameter7 in [102, 103, inf, inf]) => Parameter37=Sent email (CF = 0.98)
(Parameter11 = two) and (Parameter31 in [-inf, -inf, 16500, 20970]) and (Parameter25
in [-inf, -inf, 347, 358]) => Parameter37=Sent email (CF = 0.99)
(Parameter25 in [-inf, -inf, 34, 219]) and (Parameter7 in [-inf, -inf, 164, 192]) =>
Parameter37=Sent email (CF = 0.91)
(Parameter8 = saab) => Parameter37=Sent email (CF = 0.89)
```

```
(Parameter18 in [-inf, -inf, 516, 528]) and (Parameter25 in [335, 346, inf, inf]) =>
Parameter37=Sent email (CF = 0.88)

(Parameter11 = four) and (Parameter24 in [319, 327, inf, inf]) and (Parameter7 in [-inf, -inf, 103, 104]) => Parameter37=- (CF = 0.97)

(Parameter11 = four) and (Parameter13 = rwd) => Parameter37=- (CF = 0.95)

(Parameter15 in [945, 957, inf, inf]) and (Parameter17 in [-inf, -inf, 652, 654])
and (Parameter11 = four) => Parameter37=- (CF = 0.91)

(Parameter16 in [1736, 1768, inf, inf]) and (Parameter18 in [-inf, -inf, 555, 557])
and (Parameter18 in [548, 549, inf, inf]) => Parameter37=- (CF = 0.9)
```

### RIPPER rules:

```
(Parameter11 = four) and (Parameter24 >= 327) and (Parameter7 <= 103) =>
Parameter37=- (31.0/0.0)

(Parameter11 = four) and (Parameter13 = rwd) => Parameter37=- (25.0/1.0)

(Parameter7 <= 115) and (Parameter25 >= 327) and (Parameter17 >= 639) =>
Parameter37=- (14.0/0.0)

(Parameter11 = four) and (Parameter7 <= 91) => Parameter37=- (11.0/1.0)

(Parameter16 >= 1778) and (Parameter18 <= 555) and (Parameter12 = sedan) =>
Parameter37=- (6.0/0.0)


=> Parameter37=Sent email (118.0/7.0)
```

## References

- Agostinelli, S., Lupia, M., Marrella, A., & Mecella, M. (2020). Automated Generation of Executable RPA Scripts from User Interface Logs. In A. Asatiani, J. M. García, N. Helander, A. Jiménez-Ramírez, A. Koschmider, J. Mendling, G. Meroni, & H. A. Reijers (Eds.), *Business Process Management: Blockchain and Robotic Process Automation Forum* (Vol. 393, pp. 116–131). Springer International Publishing. [https://doi.org/10.1007/978-3-030-58779-6\\_8](https://doi.org/10.1007/978-3-030-58779-6_8)
- Aguirre, S., & Rodriguez, A. (2017). Automation of a Business Process Using Robotic Process Automation (RPA): A Case Study. In J. C. Figueroa-García, E. R. López-Santana, J. L. Villa-Ramírez, & R. Ferro-Escobar (Eds.), *Applied Computer Sciences in Engineering* (Vol. 742, pp. 65–71). Springer International Publishing. [https://doi.org/10.1007/978-3-319-66963-2\\_7](https://doi.org/10.1007/978-3-319-66963-2_7)
- Bosco, A., Augusto, A., Dumas, M., La Rosa, M., & Fortino, G. (2019). Discovering Automatable Routines from User Interaction Logs. In *Business Process Management Forum* (Vol. 360, pp. 144–162). Springer International Publishing. [https://doi.org/10.1007/978-3-030-26643-1\\_9](https://doi.org/10.1007/978-3-030-26643-1_9)
- Choi, D., R'bigui, H., & Cho, C. (2021). Candidate Digital Tasks Selection Methodology for Automation with Robotic Process Automation. *Sustainability*, 13(16), 8980. <https://doi.org/10.3390/su13168980>
- Cohen, W. W. (1995). Fast Effective Rule Induction. In *Machine Learning Proceedings 1995* (pp. 115–123). Elsevier. <https://doi.org/10.1016/B978-1-55860-377-6.50023-2>
- Fumarola, F., Lanotte, P. F., Ceci, M., & Malerba, D. (2016). CloFAST: Closed sequential pattern mining using sparse and vertical id-lists. *Knowledge and Information Systems*, 48(2), 429–463. <https://doi.org/10.1007/s10115-015-0884-x>
- Hühn, J. C., & Hüllermeier, E. (2010). An Analysis of the FURIA Algorithm for Fuzzy Rule Induction. In J. Koronacki, Z. W. Raś, S. T. Wierchoń, & J. Kacprzyk (Eds.), *Advances in Machine Learning I* (Vol. 262, pp. 321–344). Springer Berlin Heidelberg. [https://doi.org/10.1007/978-3-642-05177-7\\_16](https://doi.org/10.1007/978-3-642-05177-7_16)
- Hühn, J., & Hüllermeier, E. (2009). FURIA: An algorithm for unordered fuzzy rule induction. *Data Mining and Knowledge Discovery*, 19(3), 293–319. <https://doi.org/10.1007/s10618-009-0131-8>
- Jin, Z., & Anderson, M. R. (2017). Software for Foofah: Transforming Data by Example [Data set]. <https://doi.org/10.1145/3218889>
- Leno, V., Augusto, A., Dumas, M., La Rosa, M., Maggi, F. M., & Polyvyanyy, A. (2021a). Discovering executable routine specifications from user interaction logs. ArXiv:2106.13446 [Cs]. <http://arxiv.org/abs/2106.13446>
- Leno, V., Augusto, A., Dumas, M., La Rosa, M., Maggi, F. M., & Polyvyanyy, A. (2021b). Discovering data transfer routines from user interaction logs. *Information Systems*, 101916. <https://doi.org/10.1016/j.is.2021.101916>

- Leno, V., Augusto, A., Dumas, M., Rosa, M. L., Maggi, F., & Polyvyanyy, A.** (2020). Identifying candidate routines for Robotic Process Automation from unsegmented UI logs. arXiv:2008.05782v2. <https://arxiv.org/abs/2008.05782>
- Leno, V., Polyvyanyy, A., Dumas, M., La Rosa, M., & Maggi, F. M.** (2021). Robotic Process Mining: Vision and Challenges. *Business & Information Systems Engineering*, 63(3), 301–314. <https://doi.org/10.1007/s12599-020-00641-4>
- Leopold, H., van der Aa, H., & Reijers, H. A.** (2018). Identifying Candidate Tasks for Robotic Process Automation in Textual Process Descriptions. In J. Gulden, I. Reinhartz-Berger, R. Schmidt, S. Guerreiro, W. Guédria, & P. Bera (Eds.), *Enterprise, Business-Process and Information Systems Modeling* (Vol. 318, pp. 67–81). Springer International Publishing. [https://doi.org/10.1007/978-3-319-91704-7\\_5](https://doi.org/10.1007/978-3-319-91704-7_5)
- Manghai, A. & Jegadeeshwaran, R.** (2019). Application of FURIA for Finding the Faults in a Hydraulic Brake System Using a Vibration Analysis through a Machine Learning Approach. *International Journal of Prognostics and Health Management*, 10(1). <https://doi.org/10.36001/ijphm.2019.v10i1.2748>
- Narayana, M. B. S., Khalifa, H., & van der Aalst, W.** (2020). JXES: JSON Support for the XES Event Log Standard. ArXiv:2009.06363 [Cs]. <http://arxiv.org/abs/2009.06363>
- Razaghpanah, A., Nithyanand, R., Vallina-Rodriguez, N., Sundaresan, S., Allman, M., Kreibich, C., & Gill, P.** (2018). Apps, Trackers, Privacy, and Regulators: A Global Study of the Mobile Tracking Ecosystem. In *The 25th Annual Network and Distributed System Security Symposium (NDSS 2018)*. IMDEA. <https://dspace.networks.imdea.org/handle/20.500.12761/507>
- Soeny, K., Pandey, G., Gupta, U., Trivedi, A., Gupta, M., & Agarwal, G.** (2021). Attended robotic process automation of prescriptions' digitization. *Smart Health*, 20, 100189. <https://doi.org/10.1016/j.smhl.2021.100189>
- Syed, R., Suriadi, S., Adams, M., Bandara, W., Leemans, S. J. J., Ouyang, C., ter Hofstede, A. H. M., van de Weerd, I., Wynn, M. T., & Reijers, H. A.** (2020). Robotic Process Automation: Contemporary themes and challenges. *Computers in Industry*, 115, 103162. <https://doi.org/10.1016/j.compind.2019.103162>
- van der Aalst, W. M. P., Bichler, M., & Heinzl, A.** (2018). Robotic Process Automation. *Business & Information Systems Engineering*, 60(4), 269–272. <https://doi.org/10.1007/s12599-018-0542-4>
- Viehhauser, J., & Doerr, M.** (2021). Digging for Gold in RPA Projects – A Quantifiable Method to Identify and Prioritize Suitable RPA Process Candidates. In M. La Rosa, S. Sadiq, & E. Teniente (Eds.), *Advanced Information Systems Engineering* (Vol. 12751, pp. 313–327). Springer International Publishing. [https://doi.org/10.1007/978-3-030-79382-1\\_19](https://doi.org/10.1007/978-3-030-79382-1_19)

---

**Editorial record:** The article has been peer-reviewed. First submission received on 13 January 2022. Revisions received on 30 January 2022, 21 April 2022, and 18 June 2022. Accepted for publication on 21 June 2022. The editor in charge of coordinating the peer-review of this manuscript and approving it for publication was Stanislav Vojir .

---

Acta Informatica Pragensia is published by Prague University of Economics and Business, Czech Republic.

ISSN: 1805-4951

---