**Use of FURIA for Improving Method of Task-Mining**

Petr Průcha 1 [0000-0003-2197-7825], Jan Skrbek1

1 Faculty of Economics, Technical University of Liberec, Studentská 1402/2, 461 17 Liberec, Czech Republic; petr.prucha@tul.cz, jan.skrbek@tul.cz

**Abstract:**

Companies that use robotic process automation RPA 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 current method's limiting restrictions and use another FURIA rule learning algorithm for rule detection. We use two different datasets and the WEKA platform to validate the results. The results show that FURIA and the removal of strictly deterministic rules as limiting restrictions turned out to be the competitive approach to the original one. In this study on presented data, the selected approach detected more candidates to automation and with higher precision. This study implicates that FURIA and not using a strictly deterministic process is appropriate procedure with certain use cases as other procedures mentioned in this study.

**Keywords:** FURIA, Task-mining, RPA task-mining, RIPPER, Automatable routines

**Acknowledgment:** This paper was prepared thanks to the Technical University of Liberec SGS grant with number: SGS-2021-1033

# 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 RPA (Robotic Process Automation) 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. The comparison of the task-mining approach is described 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, other approach will be introduced and tested that could bring better results and thus improve the current algorithm from 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, where they 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 of these algorithms fall into the category of RLA algorithms (J. C. 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.

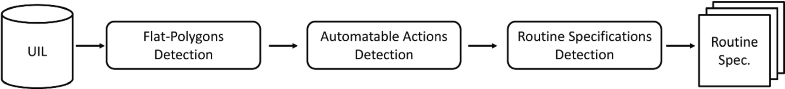
This article aims to test the possibility of using not strictly deterministic rules to identify candidates for automation and to compare the FURIA algorithm with RIPPER.

**Tab. 1**. Comparison of task-mining approaches.

|  |  |  |  |
| --- | --- | --- | --- |
| **Approach** | **Type of data/example where the principle is used** | **Are the selected candidates strictly deterministic?** | **Authors** |
| They use strictly deterministic rules. They use the Foofah algorithm to discover the rules (Jin & Anderson, 2017). | UI logs – They try to find rules for transforming data between Excel tables. | Yes | (Leno, Augusto, et al., 2021a) |
| They try to find the formula in the data with the most similarity and based on that. They generate routines for automation. They use PM4PY and a-priori. | UI logs – Move data from an Excel spreadsheet to a web form. | No | (Agostinelli et al., 2020) |
| They represent 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) |
| They present methods for evaluating candidates for automation based on selected factors, which are evaluated using marks. The resulting marks serve as a suitability evaluation factor. | Process logs (not exactly specified) - Tested on nine different examples. | No | (Viehhauser & Doerr, 2021) |
| They discover routines and sequences that are strictly deterministic, use of RIPPER, FOOFAH algorithms. | UI logs – Rewriting input data from the study department into a web form and Excel. | Yes | (Bosco et al., 2019) |
| Selection and discovery routines based on the CloFast algorithm (Fumarola et al., 2016). They 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 a web form and an Excel spreadsheet. | Yes | (Leno, Augusto, et al., 2021b) |
| Using the AI to process the so-called NPL based on text inputs predicts whether the activity is automatable. | Process logs– 47 different processes from 10 different sources | No | (Leopold et al., 2018) |

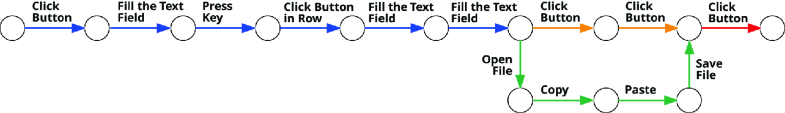
## Bosco approach

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, see fig. 1. In the first phase, they have sorted UI logs from the process. They then detect 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 fig. 2. After finding all process paths, it is detected whether the given action is automated, including all parameters that the given action has. 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 as 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. One to find rules in the data is called RIPPER. The second algorithm is called FOOFAH and is used to find rules for data transformation. It is used mainly in connection with spreadsheets.



**Fig. 1.** Bosco 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 for their routine 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.



**Fig. 2**. Working example in Bosco article. Source (Bosco et al., 2019)

### Bosco working example

An example in which Bosco et al., (2019) test its approach can be seen in fig. 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 fills in the information in the information system. Subsequently, if the student is Australian, he will confirm his address, which the study department backs up to Excel (green variant). If it 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 filling by clicking a button.

## Approach

As indicated in the introduction of the article, Bosco et al., (2019) approach are 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 from 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 part of a process, and then the robot completes the activities, or another possibility is that the robot cannot process, mark, and continues, 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, Polyvyanyy, 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. From which it can be assumed 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, the rule has been rewritten into pseudocode. The accuracy of the FURIA algorithm rules is 50%.

If paramenter36 == 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, which are in tab. 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 by candidates who are not strictly deterministic, as we assume that at least part of the process can be automated, even if we do not know the rules that would meet 100% of all cases.

# Methodology

This research, as already written, follows the procedures of Bosco et al., (2019), where other methods are used to determine whether the current approach can be improved. In this research, we focus on only 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 Chapter 1.1. 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 as it follows from the literature search and communication with RPA specialists, 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 Chapter 1.1. 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 from log6.mxlm (Bosco et al., 2019) and the UI logs presented in Chapter 2.1. The WEKA (Waikato Environment for Knowledge Analysis) platform version 3.8.5, which was installed on a Windows 10 computer, was used to calculate and search for rules. The WEKA platform integrates RLA algorithms, RIPPER (jRip), and FURIA. The basic settings in the WEKA program are used to determine the rules.

Data[[1]](#footnote-1) log6.mxlm has been converted to CSV format using this program and can be found in the repository under the name Log6Result.csv. An example of transformed data named Log6Result.csv is in tab. 2 in transformed data is not second line of tab. 2. Original MXML format of data is version of XML. In process mining field is used hojně využíván jiný formát XML, který se nazývá XES. XES format is process mining standardes format based on IEEE standards. XES je vhodný pro přenášení event logů, protože věrně zachycují realitu. XES také podporuje nejvíce process mining nástrojů. (Narayana et al., 2020) Ikdyž MXML a XES jsou formáty XML, tak XML formát není vhodný pro další analýzy a proto process-mining tools nabízejí transformace XML do CSV. Převedený MXML do CSV bylo provedeno následovně.

MXML log je tvořen z AuditTrailEntry, který je následně tvořen hlavním tagem <Data> a dalšími komponenty, které se nachází za tagem </Data>. Pro náš výzkum nás zajímala především hodnoty uvnitř tagů <Data> a </Data>. Hodnotu attribute name jsme použili abstraktně jako název sloupce a hodnotu uvnitř tagu attribute jsme použili jako hodnotu pro daný sloupec. Každá další data akce v AuditTrailEntry pro související procesní záznam jsou řazena do následujícího sloupečku. Each line indicates one process record, which consists of multiple actions. Throughout the research, this format/approach is used. Každý sloupec je označen jako parameter a číslo daného sloupce. Hodnoty attribute name nejsou použity, jelikož nenesou hlavní informaci a pro náš výzkum byla použita essential data. Navíc by bylo velmi nepřehledné využívat pro názvy sloupců, hodnoty, které se opakují. Automatická konverze například v process-mining nástroji DISCO, využije hodnotu attribute name a hodnotu tagu a spojí je dohromady pomocí znaku „ : “. Takže hodnoty první tři hodnoty by byly z příkladu by vypadali následovně:

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

Ukázka formátu MXML:

<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>

The actions are broken down into parameters and in the same sequence as the original format. Each line indicates one process record, which consists of multiple actions. Throughout the research, this format/approach is used for simplification and only works with distributed parameter actions.

Parameter 36 will be tested for Log6Result.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 for these remaining parameters would be uniquely determined to be 100% except for the value for Timestamp and parameter 51. All tested parameters are in tab. 4.

**Tab. 2**. Sample data – log6.csv. Source transformed data 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 chapter 2.1., The data is already in CSV format, so no further transformation is required. For this data, parameter 37 has the most significant value because the remaining parameters are identical, or it is input data, see parameter 7 to parameter 31.

**2.1 Working example**

Furthermore, data was 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. Data[[2]](#footnote-2) was 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 is prepared for entering new data or ended in the previous step. Visualization process see fig. 3.

**Graphical user interface, text, application

Description automatically generatedFig. 3.** Working example of the process for our data. Source: authors

# Results

The results obtained when testing parameter 36 on the data named Log6Result.csv can be seen in tab. 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 have determined all instances correctly, thanks to the rules below.

**Tab. 3.**Algorithm results on Log6Result.csv data.

|  |  |  |
| --- | --- | --- |
|  | **FURIA** | **RIPPER** |
| **Precision** | 100 % | 100 % |
| **Correctly Classified Instances** | 999 | 999 |
| **Incorrectly Classified Instances** | 0 | 0 |

The rules discovered in the Log6Result.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 on 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/

In tab. 4 are the accuracy results of all the rules that the algorithms were able to find for the given parameter. All parameters were selected by rules stated in chapter 2. In most cases, the accuracy of the algorithms is the same except for parameter42, where FURIA is 0.7% more accurate than RIPPER. FURIA was able to identify 7 cases more than RIPPER correctly. Which point out better performance of FURIA Parameter42 is consist of identification about user (value) and identification of platform (source). All these results are rounded to one decimal place.

**Tab. 4.** Results of comparison of RIPPER and FURIA in Log6Result.csv data.

|  |  |  |
| --- | --- | --- |
|  | **FURIA** | **RIPPER** |
| **Parameter8** | 8.3 % | 8.3 % |
| **Parameter19** | 8.3 % | 8.3 % |
| **Parameter25** | 0 % | 0 % |
| **Parameter30** | 49.8 % | 49.8 % |
| **Parameter36** | 100 % | 100 % |
| **Parameter37** | 100 % | 100 % |
| **Parameter38** | 100 % | 100 % |
| **Parameter39** | 100 % | 100 % |
| **Parameter40** | 49.8 % | 49.8 % |
| **Parameter41** | 50.2 % | 50.2 % |
| **Parameter42** | 51.4 % | 50.7 % |
| **Parameter43** | 100 % | 100 % |
| **Parameter44** | 52.3 % | 52.3 % |
| **Parameter45** | 49.8 % | 49.8 % |
| **Parameter46** | 100 % | 100 % |
| **Parameter47** | 100 % | 100 % |
| **Parameter48** | 50.2 % | 50.2 % |
| **Parameter49** | 100 % | 100 % |
| **Parameter50** | 49.8 % | 49.8 % |

The following results were found in the data from chapter 2.2 entitled Auto2Mail.csv, see tab. 5.

**Tab. 5.**Result of sample data Auto2Mail.csv.

|  |  |  |
| --- | --- | --- |
|  | **FURIA** | **RIPPER** |
| **Precision** | 88,8 % | 80 % |
| **Correctly Classified Instances** | 182 | 164 |
| **Incorrectly Classified Instances** | 23 | 41 |

On these data, the rules of the FURIA algorithm are clearly more precise than RIPPER. The rules for these cases can be found in Appendix A. FURIA correctly identified 182 cases out of 205 cases, and 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 this data. In tab. 5, the results are rounded to one decimal place.

# 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 its 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. Problem with not default monitor is that 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 that a person makes can be very difficult to detect by artificial intelligence. (Leno, Augusto, 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 automated 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 the 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. (J. Hühn & Hüllermeier, 2009; Manghai & Jegadeeshwaran, 2019). Therefore, the results of our research have certain limitations associated with the dataset used. Therefore, it is appropriate for future research to test this approach on other data that will provide a more comprehensive view of this approach.

## 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 RLA FURIA algorithm has been shown to be better than RIPPER on test data, however, as other experts mention, the data is 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 non-strictly deterministic and using the FURIA algorithm is beneficial.

**Bibliografie**

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

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

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

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. Wierzchoń, & 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

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*.

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

Manghai, A. & Jegadeeshwaran. (2019). *Application of FURIA for Finding the Faults in a Hydraulic Brake System Using a Vibration Analysis through a Machine Learning Approach*. *20*, 9. http://www.phmsociety.org/node/2614

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, February 18). *Apps, Trackers, Privacy, and Regulators: A Global Study of the Mobile Tracking Ecosystem*. The 25th Annual Network and Distributed System Security Symposium (NDSS 2018). 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

**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)

1. Data is possible download here: https://github.com/Scherifow/SGS-Task-Mining [↑](#footnote-ref-1)
2. Data is possible to find in repository: https://github.com/Scherifow/SGS-Task-Mining [↑](#footnote-ref-2)