PUSL 2076

Data Programming in R

Research Paper

Group Member Details

PU Index No.	Student Name	Degree Program
10899483	Fernando Naveen	BSc (Hons) Data Science
10899178	Konganige N Anthony	BSc (Hons) Data Science
10899495	Thehara Weerasiri	BSc (Hons) Data Science
10899488	Senanayake Senanayake	BSc (Hons) Data Science
10899177	Urulugastenne Amarakone	BSc (Hons) Data Science

Student Enrollment Process Optimization Using Process Mining

Senanayake Senanayake[1], Fernando Naveen [2], Konganige N Anthony[3], Urulugastenne Amarakone[4], Thehara Weerasiri[5]

[12345]Undergraduate, BSc (Hons) Data Science.

Abstract

To extract useful insights from the student enrollment process, this study explores the domain of process mining approaches, concentrating on determining crucial and ideal pathways. Through the creation of process maps, the research attempts to uncover hidden patterns present in enrollment data using heuristic mining methods. By Using heuristic mining, the project seeks to reveal hidden avenues in the enrolling process, providing educational institutions with useful information. In addition, the study investigates the application of Dijkstra's algorithm to identify optimal pathways, providing insight into the most effective methods from start to finish. Using these initiatives, the research aims to improve the enrollment process for students, providing educational institutions with a deep comprehension of enrollment dynamics and enabling well-informed decision-making procedures.

1.Introduction

In the ever-changing world of education, organizations are under increasing pressure to streamline administrative processes successfully meet the needs of a wide range of students. Enrolling new students is a crucial activity that marks the beginning of their academic journey. Understanding the importance of this, our research uses process mining tools to try and decipher the complexities of student enrollment. Our objective is to improve decision-making and optimize operations in this crucial area by detecting latent patterns and utilizing insights derived from data. As the foundation of data science, process mining has great potential in this endeavor, providing ways to spot bottlenecks, foresee problems, and suggest tactical actions. Our work highlights the usefulness of process mining by validating its results using, among other things, the Heuristic Miner algorithm and real-world case studies.

2. Literature Review

A. Rozinat and W. M. P. van der Aalst's "Process Mining Algorithms: A Systematic Review" (2006) provides a comprehensive analysis of process mining algorithms, with an emphasis on Heuristics Miner in particular. The study assesses different algorithms according to performance metrics and efficacy, offering insightful information about the advantages and disadvantages of each. By contrasting these algorithms, it advances knowledge of process mining methodologies and helps professionals choose the best strategy for their particular requirements. By pinpointing weaknesses and opportunities for enhancement in currently available algorithms, Rozinat and van der Aalst's systematic review establishes a strong basis for future investigations in the domain. [1]

"Conformance Checking of Processes Based on Monitoring Real Behavior" by W. M. P. van der Aalst et al. (2008) explains the idea of conformance checking, which is important for determining how well processes' modeled and observed behavior align. The application of Heuristics Miner and other techniques to conformance checking is discussed in the paper, offering insightful information on evaluating process compliance. This research helps to improve the accuracy and effectiveness of process management systems, which in turn improves organizational performance and decision-making processes, by comparing real-world behavior to process models. [2]

Heuristic Miner for Performance Perspectives (HMp) is an extension of Heuristics Miner, presented in "Discovering Workflow Performance Models from Timed Logs" by M. Weidlich et al. (2009). This module addresses the need for assessing process performance in addition to structure, with a focus on mining performance-related data from timed logs. Organizations can uncover areas for optimization and gain deeper insights into their operational efficiency by using HMp, which boosts process discovery and analysis capabilities by integrating time-related data into the process mining framework. [3]

The method of mining process models from workflow logs is explored in "Mining Process Models from Workflow Logs" by W. M. P. van der Aalst et al. (2011), who emphasize the importance of Heuristics Miner in this particular context. In addition to discussing issues with noise and incompleteness in log data, the paper provides case examples that demonstrate how process discovery approaches are used in real-world scenarios. This research advances the subject of process mining and offers helpful advice for businesses looking to enhance their business processes by tackling these issues and presenting practical examples. [4]

3.Methodology

3.1Introduction to the event log dataset

The given dataset is a subsample of event log data on the student enrollment process. Each row in the dataset refers to a particular event on a student enrollment case. The dataset consists of several attributes: ‘Case_ID’, ‘Enrollment_Status’, ‘Active_Status’, ‘Reassignment_Count’, ‘Reopen_Count’, …etc. These attributes contain information about a given event such as its status, timestamps of handling, actors, and other relevant indicators. Evidently, the dataset captures several stages of the student enrollment process from cases that are opened, updated, resolved, and closed. It is possible to analyze this data based on patterns and trends as well as anomalies to understand the efficiency, performance, and compliance of the process. In this regard, stakeholders will learn more about the current process, weaknesses to address, and improvements to implement for a better student enrollment experience.

3.2 Usage of libraries

In order to handle event log data to perform various process mining tasks and analysis, there are important libraries in R which can be used for these purposes.

1. **bupaR** - This library in R has been used to import and prepare the data for analysis.
2. **eventdataR** – This library in R has been used to create and handle the data structure of the event log.
3. **ProcessmapR** – This library in R has been used to create the process map based on the event log which depicts the sequence of events.
4. **edeaR** – This library in R has been used to manage functions of event log analysis which includes filtering and pattern recognition.
5. **dplyr** – This library in R has been used for data preprocessing, organizing and summarizing data.

3.3 Preprocessing

3.3.1 TimeStamp

The "Last_Updated_At" column in the dataset is the most suitable for process mining, as it indicates the moment when the enrollment process underwent a significant update or change. The data type was changed to POSIXct format for consistency and compatibility with R's analysis tools, enabling precise time-based analysis, duration computations, and visualization of process behavior.

3.3.2 Missing values

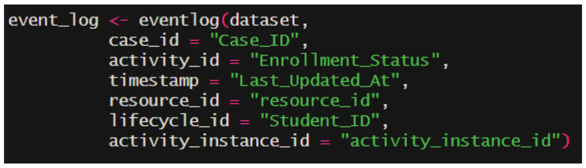
In the second stage of data preprocessing, we went through the dataset to look for any missing values. We paid special attention to the columns Case_ID, Enrollment_Status, Last_Updated_At, and Student_ID, which are essential for generating the event log. After careful examination, none of these crucial columns had any missing data. As a result, handling missing values in this dataset or using imputation techniques was not necessary. This result emphasizes how reliable and robust the dataset is, which guarantees the accuracy and consistency of any further analysis carried out for process mining.

3.3.3 Removing “-100”

One data item that we found in the Enrollment_Status column (designated by "-100") did not match any legitimate enrollment process activity. This was discovered during the data preprocessing stage. Taking preemptive steps to eliminate this anomaly from the Enrollment Status column, we acknowledged the significance of preserving data integrity. We assure the dataset's quality and dependability by removing this incorrect entry, which guards against any potential distortions or mistakes in further analyses—especially when it comes to process mining. Our dataset is of higher quality because of this careful data-cleaning method, which also makes it easier to derive deeper conclusions about the enrolling process.

3.4 Creating the event log

As a primary practice in process mining analysis, event logs are used to input the data required in the process mining analysis. So this will arrange and organize the data in a suitable format which is in the standard format and ready for doing process mining.To create the event log, eventlog() function was used from the eventdataR library. In the eventlog function there are special parameters namely ‘case_id’, ‘activity_id’, ‘timestamp’, ‘resource_id’ , ‘lifecycle_id’ and ‘activity_instance_id’. These parameters are initialized with the corresponding columns in the event log dataset. ‘case_id’ is the parameter which is used to specify the column which uniquely identifies the process instances in the event log dataset. ‘activity_id’ is the parameter which includes the column that represents the action performed as a part of the process. ‘timestamp’ is the parameter which includes the column that has can be used to identify the time each event occurred. ‘resource_id’ is the parameter which includes the column that has the data of which person or resource is responsible for each event. ‘lifecycle_id’ is the parameter which includes the identifier of each instance in the event log data. ‘activity_instance_id’ is the parameter which includes the parameter that uniquely identifies each instance in the event log data. When the event log is created with the relevant columns for the above-mentioned parameters, process mining is done based on this structure.

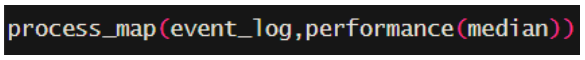


```
event_log <- eventlog(dataset,
          case_id = "Case_ID",
          activity_id = "Enrollment_Status",
          timestamp = "Last_Updated_At",
          resource_id = "resource_id",
          lifecycle_id = "Student_ID",
          activity_instance_id = "activity_instance_id")
```

Figure 1 Creating the Event Log

3.5 Creating the process map

Process maps are created based on the event log created previously. In order to create process maps, the processmap() function which is available in the ‘processmapR’ library is used. As parts of the function parameters such as event_log and parameter are used. As the ‘event_log’ parameter, we include the event log which includes the structure to visualize the process map. ‘performance’ parameter is used to specify how the performance of the activities should be analyzed in the event log. In this scenario it is specified as ‘median’ which means the median performance of the activities is analyzed when visualizing the process map. The output of this function ‘processmap’ is a diagram as stated above. In this diagram, nodes with activities connected to through edges representing the flow of activites in the event log data.



```
process_map(event_log,performance(median))
```

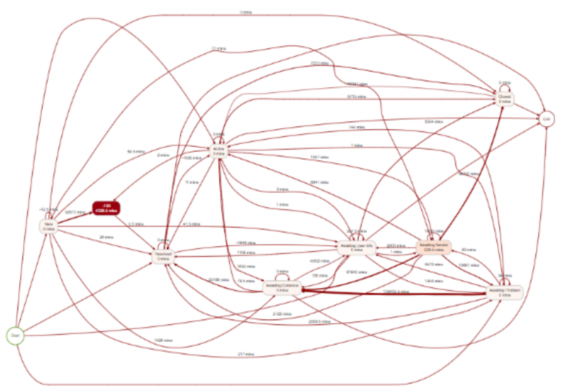


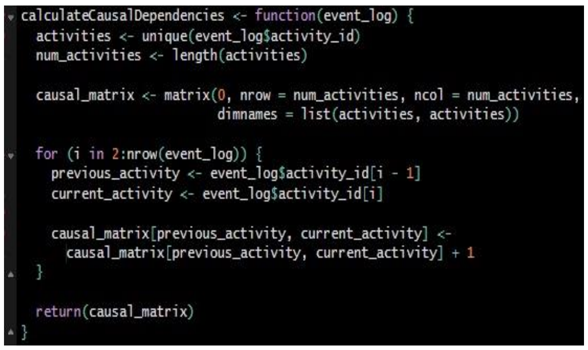
Figure 2 Process Map Before Using the Mining Algorithm

3.6 Heuristic Miner Algorithm

We decided to implement the Heuristic Miner algorithm as our process mining technique because it is a good fit for identifying process models from event data, especially in situations when the underlying process isn't explicitly described or fully structured. In line with the properties of our dataset and analytical goals, the Heuristic Miner algorithm has many benefits. Heuristic Miner is a great tool for analyzing enrollment processes because it can handle event logs with a variety of complex and diverse process behaviors. These processes frequently involve several paths and decision points. Heuristic Miner also performs exceptionally well at identifying processes from the control flow and performance viewpoints, giving us insights into the order, frequency, and duration of the various operations.

The process of applying the Heuristic Miner algorithm to an event log dataset for process mining purposes. The process involves several key steps:

1. **Causal Dependency Calculation** The ‘calculateCausalDependencies’ function calculates a causal matrix to represent the relationships between activity pairs in the event log. The count of transitions between activities is increased by iterating over the event log and initializing a matrix with zeros.

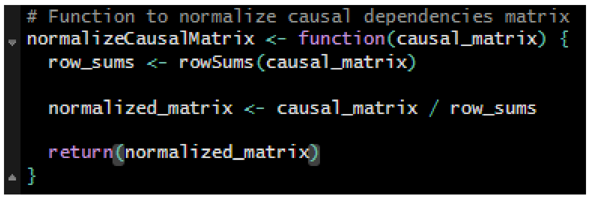


```
calculateCausalDependencies <- function(event_log) {
  activities <- unique(event_log$activity_id)
  num_activities <- length(activities)

  causal_matrix <- matrix(0, nrow = num_activities, ncol = num_activities,
                          dimnames = list(activities, activities))

  for (i in 2:nrow(event_log)) {
    previous_activity <- event_log$activity_id[i - 1]
    current_activity <- event_log$activity_id[i]

    causal_matrix[previous_activity, current_activity] <-
      causal_matrix[previous_activity, current_activity] + 1
  }

  return(causal_matrix)
}
```

Figure 3 Casual Dependency Calculation

2. **Normalization**

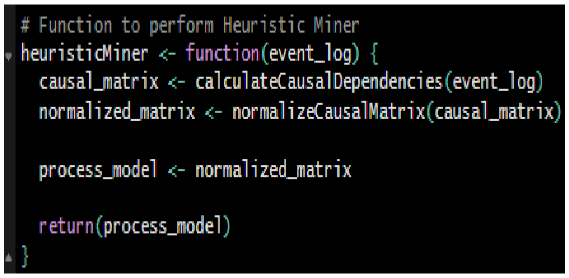
The ‘normalizeCausalMatrix’ function divides every row in the causal matrix by its sum to normalize it. By taking this step, you can be sure that every row accurately depicts the likelihood of a transition from one activity to another.



```
# Function to normalize causal dependencies matrix
normalizeCausalMatrix <- function(causal_matrix) {
  row_sums <- rowSums(causal_matrix)

  normalized_matrix <- causal_matrix / row_sums

  return(normalized_matrix)
}
```

Figure 4 Normalization.

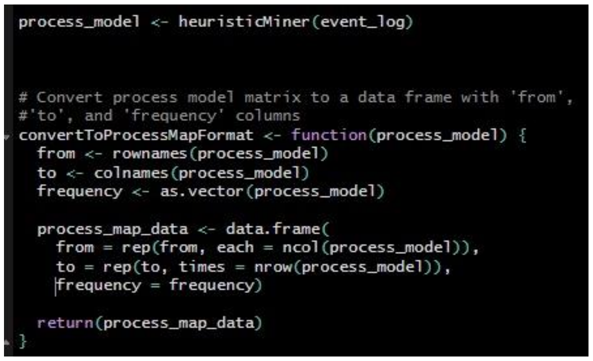
3. **Heuristic Miner Algorithm** The ‘heuristicMiner’ function utilizes the event log data and the Heuristic Miner technique. To generate a process model, it computes and normalizes the causal linkages. The identified process flow, which is based on the observed transitions between activities, is represented by the process model.



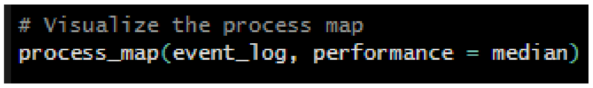
```
# Function to perform Heuristic Miner
heuristicMiner <- function(event_log) {
  causal_matrix <- calculateCausalDependencies(event_log)
  normalized_matrix <- normalizeCausalMatrix(causal_matrix)

  process_model <- normalized_matrix

  return(process_model)
}
```

Figure 5 Heuristic Miner

4. **Conversion to Process Map Format** The ‘convertToProcessMapFormat’ function transforms the matrix of the process model into a data frame format that can be visualized. From the process model matrix, it extracts the row and column names (activities) and their accompanying frequencies, arranging them into a data frame with columns labelled "from," "to," and "frequency".



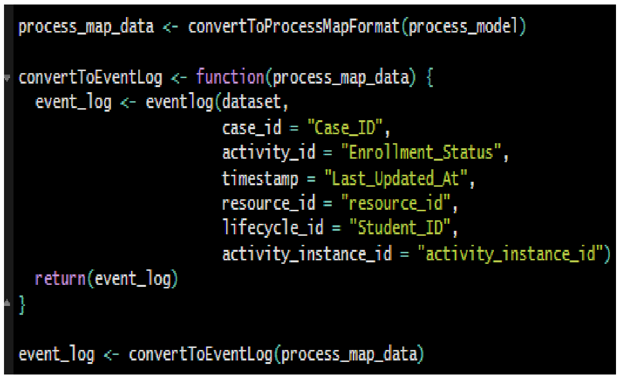
```r
process_model <- heuristicMiner(event_log)


# Convert process model matrix to a data frame with 'from',
#'to', and 'frequency' columns
convertToProcessMapFormat <- function(process_model) {
  from <- rownames(process_model)
  to <- colnames(process_model)
  frequency <- as.vector(process_model)

  process_map_data <- data.frame(
    from = rep(from, each = ncol(process_model)),
    to = rep(to, times = nrow(process_model)),
    frequency = frequency)

  return(process_map_data)
}
```



```r
# Visualize the process map
process_map(event_log, performance = median)
```

Figure 6 Conversion to Process Map Format

5. **Conversion to Event Log**
   The 'convertToEventLog' function transforms the process map data frame back into an event log format, which is necessary for process mining tools' visualization and additional analysis.



```r
process_map_data <- convertToProcessMapFormat(process_model)

convertToEventLog <- function(process_map_data) {
  event_log <- eventlog(dataset,
                        case_id = "Case_ID",
                        activity_id = "Enrollment_Status",
                        timestamp = "Last_Updated_At",
                        resource_id = "resource_id",
                        lifecycle_id = "Student_ID",
                        activity_instance_id = "activity_instance_id")
  return(event_log)
}

event_log <- convertToEventLog(process_map_data)
```

Figure 7 Conversion to Event Log

6. **Visualization**
   Lastly, performance metrics like median duration can be shown when the process map is visualized using the process_map function from the processmapR library.The technique described above makes the ability to use the Heuristic Miner algorithm to examine event log data and extract knowledge from the process map that is produced. Using the Heuristic Miner technique, we may find patterns and dependencies in the process flow that the event log depicts. We can examine the process map that is produced after the algorithm has been executed and the process model is obtained to extract useful data.

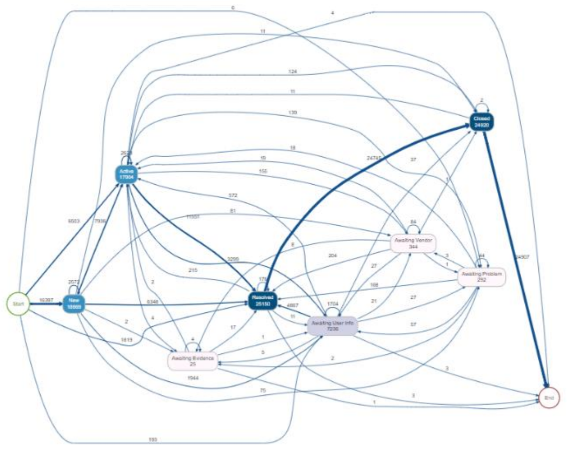
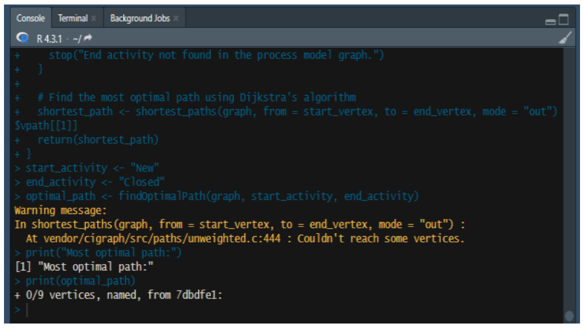


Figure 8 Process Map After Using Heuristic Miner

4. Evaluation

The Dijkstra's algorithm can be used to find the optimal path of a generated process map, but it cannot fully rely on code implementation. In the case of student enrollment event log, the algorithm's complexity prevents reaching certain vertices.



```
+     stop("End activity not found in the process model graph.")
+   }
+
+   # Find the most optimal path using Dijkstra's algorithm
+   shortest_path <- shortest_paths(graph, from = start_vertex, to = end_vertex, mode = "out")
$vpath[[1]]
+   return(shortest_path)
+ }
> start_activity <- "New"
> end_activity <- "Closed"
> optimal_path <- findOptimalPath(graph, start_activity, end_activity)
Warning message:
In shortest_paths(graph, from = start_vertex, to = end_vertex, mode = "out") :
  At vendor/cigraph/src/paths/unweighted.c:444 : Couldn't reach some vertices.
> print("Most optimal path:")
[1] "Most optimal path:"
> print(optimal_path)
+ 0/9 vertices, named, from 7dbdfe1:
> |
```

Figure 9 Optimal Path Code Output

Because of this issue we can analyze the process map and consider the optimal path for the student enrollment process.In the process map we obtained through heuristics miner, there is more than one path we can take to complete the enrolment process. But one path stands out the most. After starting the process algorithm will choose new or active problem. Then it will wait for problem info. After that problem will be resolved and closed. However, we can’t say for sure that this is the most critical path to complete this because decencies of the events cannot be identified clearly.

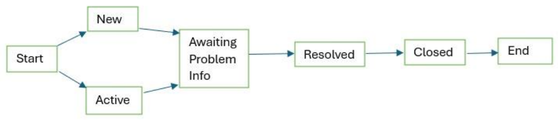


Figure 10 Optimal Path Using Process Map

After starting the process algorithm will choose new or active problem. Then it will wait for problem info. After that problem will be resolved and closed.

5. Conclusion

Even though Dijkstra's method proved to be difficult to implement, our investigation of process mining approaches in the context of the student enrolling process yielded insightful findings. We found several paths using heuristic mining, but one path stood out as being frequently taken. Dependencies between events, however, cast doubt on how crucial they are.

Further research is necessary to properly understand the causal links behind the enrolling process, even though it provides some initial insights. For increased accuracy, process mining algorithms may be improved and new data sources may be used in future studies. The present study highlights the significance of process mining in enhancing intricate organizational procedures such as student enrolment, hence promoting gains for both students and administrators.

References

[1] A. R. a. W. M. P. v. der Aalst, "Process Mining Algorithms: A Systematic Review," [Online]. Available: https://www.researchgate.net/publication/225389374_Discovering_Workflow_Performance_Models_from_Timed_Logs .

[2] W. M. P. v. d. Aalst et al., "Conformance Checking of Processes Based on Monitoring Real Behavior," [Online]. Available: https://www.sciencedirect.com/science/article/pii/S030643790700049X.

[3] M. Weidlich et al., "Discovering Workflow Performance Models from Timed Logs," [Online]. Available: https://www.researchgate.net/publication/225389374_Discovering_Workflow_Performance_Models_from_Timed_Logs.

[4] W. M. P. v. d. Aalst et al., "Mining Process Models from Workflow Logs," [Online]. Available: https://www.researchgate.net/publication/235950074_Process_Mining_Overview_and_Opportunities.