University of Camerino

School of Science and Technologies

Master Degree in Computer Science (LM-18 Class)
Process Mining Course

Process Mining Report

Eduardo Tiano
Student ID 129508

A.Y. 2024/2025

1. Abstract

Nowadays, Process Mining has become one of the most used techniques to analyse event logs generated by information systems.

This project presents the analysis of a real-life event log, containing records of sepsis cases from a hospital, using the Disco Process Mining tool, together with mining performed by two algorithms illustrated during the course.

The objectives are:

- to study how a process discovery algorithm works within the Disco tool, and report the insights obtained analysing the selected scenario;
- explore the differences between Alpha Miner and Heuristic Miner algorithms using the PM4PY library.

2. Log structure

2.1 General description

The log used in this project originates from an online dataset repository [3], and contains data from a real-life scenario involving sepsis cases in a hospital.

The events were recorded by the hospital's ERP system and capture over 15,000 events related to 16 distinct activities. Each event is associated with a case (or trace), representing a patient, and includes a timestamp and several attributes such as the activity name, lifecycle transition, resource, and organisational role.

All data have been anonymised to preserve patient privacy while maintaining consistency and coherence within each trace.

2.2 File format

The log is provided in the *eXtensible Event Stream* (XES) format [7], which is the de facto standard for storing and exchanging event logs in the field of Process Mining. XES is an XML-based format that supports structured metadata, including standard extensions for timestamps, resources, organisational information, and lifecycle transitions. This structure enables advanced analysis by allowing each event to be contextualised within its process instance and broader organisational framework.

2.3 XES structure

The structure of an XES file is formally described by a UML class diagram defined by the XES standard. At the core of the model is the `Event` class, which is linked to the `Trace` class, representing a complete case or process instance. Each `Trace` contains a sequence of `Event` objects, and both can include a set of `Attributes` that provide contextual information, such as the activity name (`concept:name`), timestamp (`time:timestamp`), and lifecycle transition (`lifecycle:transition`).

The UML diagram defines the hierarchical relationship between classes like `Log`, `Trace`, `Event`, and `Attribute`, ensuring the format is both machine-readable and semantically rich. This design facilitates consistent parsing, validation, and interpretation of event data across different process mining tools.

3. Project

3.1 Disco tool

Disco is a process mining tool developed by Fluxicon, designed to support interactive analysis of event logs. It is widely recognised for its ease of use and efficiency in handling real-life process data. Disco utilises the Fuzzy Miner algorithm, well-suited for discovering models even from not-well-structured logs.

Disco accepts logs in the XES format, and automatically detects the case identifier, activity names, and timestamps during import. Once the log is loaded, the tool generates a dynamic process map that can be explored and simplified interactively by adjusting the level of detail using a slider. This enables the analyst to focus on the most frequent behaviour or reveal exceptional paths, depending on the objective.

In addition, Disco provides filtering capabilities, allowing users to define subsets of the log based on time frames, cases, paths, performance metrics, and other attributes. These filters are essential for comparative analysis and for isolating specific process variants or bottlenecks.

The tool also includes statistical views, such as frequency and performance dashboards, which offer insights into activity durations. Although Disco does not explicitly calculate fitness and precision in a formal manner, it offers a practical and visually rich environment for process analysis, making it particularly valuable for business analysts and practitioners.

3.2 Fuzzy Miner algorithm

The algorithm splits event logs into smaller sub-logs, creates a directly followed graph, showing the relationships between activities in the sub-log, and uses these metrics to identify the most relevant activities.

In the end, it is also possible for the user to adjust the level of detail in the process model.

The main advantages are that it can handle more easily unstructured logs (containing noise), and focusses on essential patterns.

The main limitations are represented by the fact that it's not directly convertible to other process modelling languages, and the results could be difficult for the user to interpret sometimes.

3.3 Analysis of the log

Once the log is uploaded in Disco, in the map section of the software it's possible to observe the flow of activities: the thickness of the arrows and the colour intensity blocks reflect how often each path and activity occurred.

For instance, it is clear that the most common path is represented by the *ER Registration* activity, followed by *ER Triage*, *ER Sepsis Triage*, and so on. Thin edges show uncommon paths. It is worth mentioning that *CRP* and *Leucocytes* are heavily repeated steps. This is show in image 3.1.

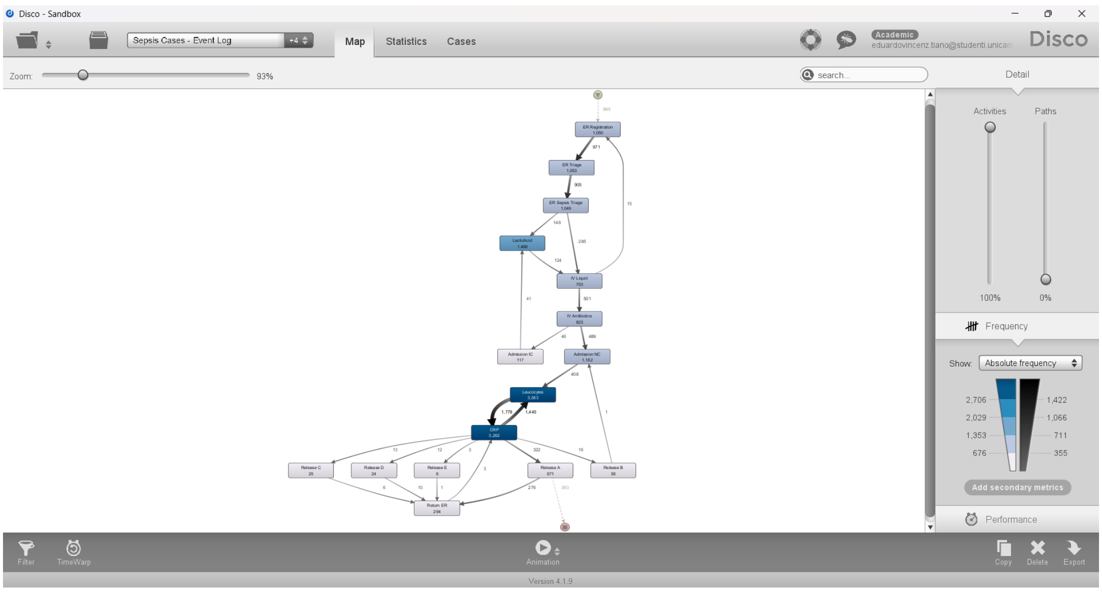

Figure 3.1: Main page in Disco.

It is also possible to visualise an arbitrary number of activities, as show in figure 3.2. Even though it's not shown in the picture, there's an option to visualise more (or less) paths, allowing the user to focus on different perspectives.

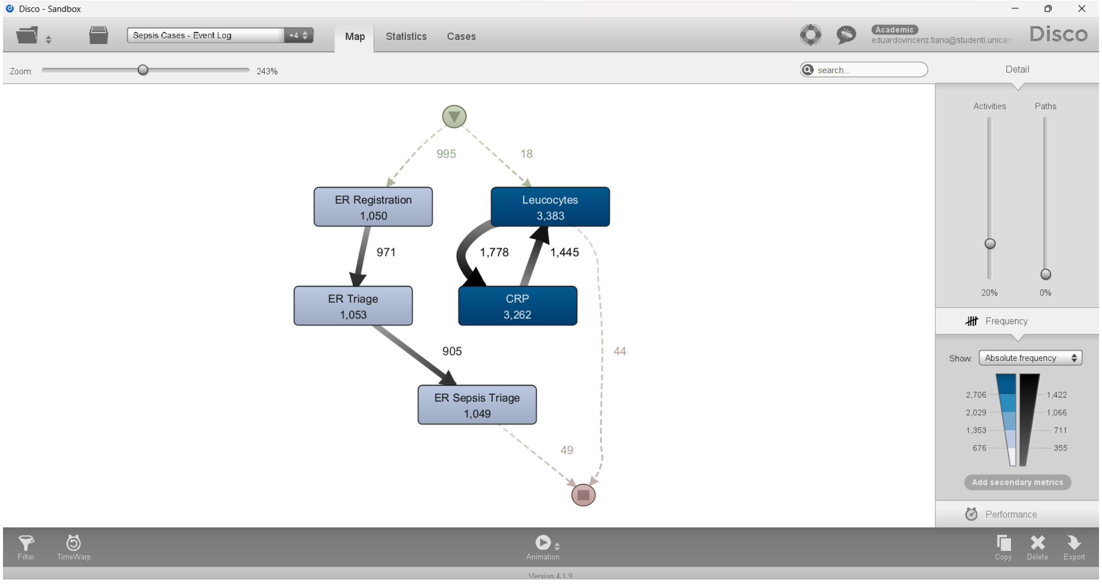

Figure 3.2: Main page in Disco showing an arbitrary number of activities.

It is also possible to visualise the case coverage instead of the absolute frequency, as well as other parameters, as shown in figure 3.3. The software is equipped with options to obtain insights regarding performance too, such as average time between activities, useful to detect bottlenecks.

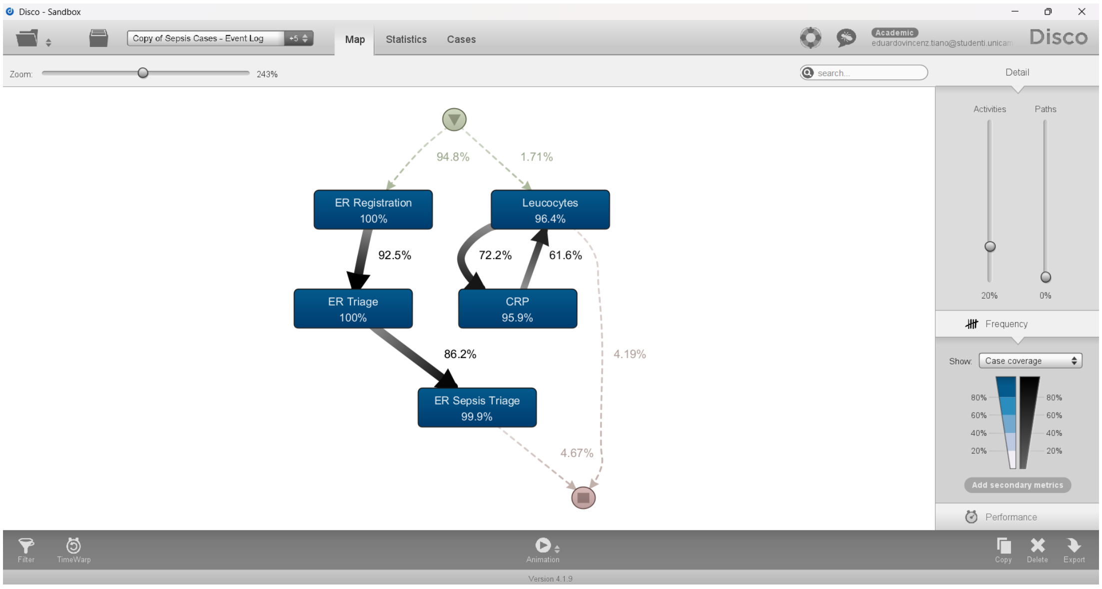

Figure 3.3: Case coverage perspective.

The tool also offers a complete visualisation of variants, containing statistics related to them, such as frequency, number or activities, and provides a visual representation as well, as shown in figure 3.4.

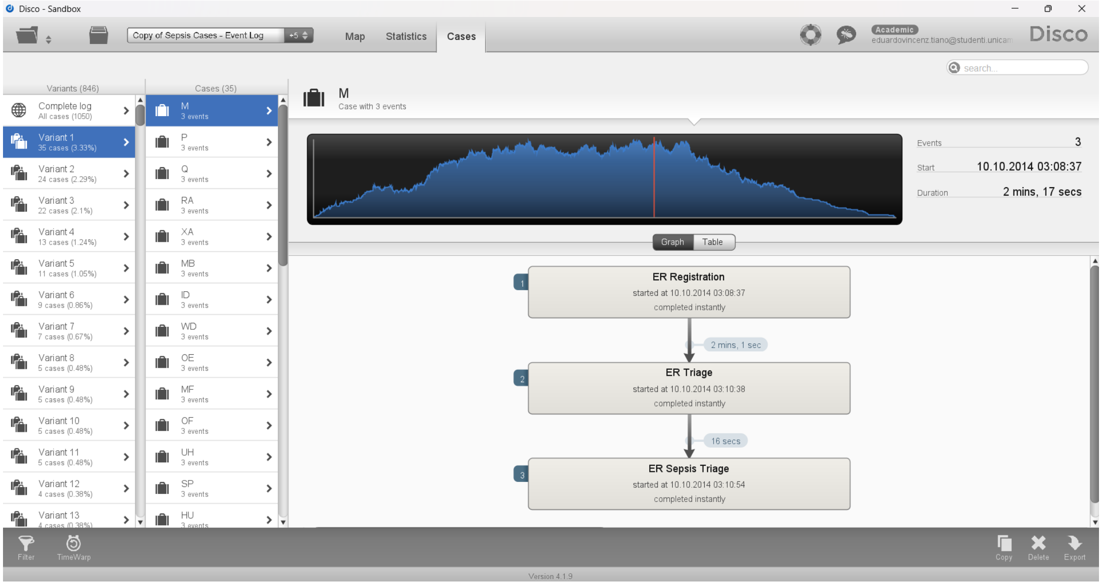

Figure 3.4: Perspective on variants.

3.4 PM4PY, Heuristic Miner and Alpha Miner

PM4PY is a Python library widely used in the Process Mining field. It allows users to execute Process Mining algorithms on logs. In this case, two algorithms have been executed on the considered log: Alpha Miner and Heuristic Miner.

The first is the oldest, and is based on identifying order relations between activities. It's quite sensitive to noise, and this makes it unsuitable for many real-life logs. The main limitations consist in the fact that it can't discover loops of length 1 and 2, may generate implicit places, and isn't able to discover non-local dependencies.

The second represents an evolution of the first, and is noise tolerant. It uses frequency-based thresholds to discover heuristic nets. It support optional paths and is widely used in real-life scenarios.

3.5 Application

The executed code is available in this GitHub repository [8]. The obtained results are the following:

- **Alpha Miner:**
  - Average Fitness:
  - Log Fitness:
  - Percentage of Fitting Traces:
- **Heuristic Miner:**
  - Average Fitness:
  - Log Fitness:
  - Percentage of Fitting Traces:

Even though both algorithms have reached quite poor results, it is worth noticing that the Heuristic Miner performed better compared to the Alpha Miner, a higher average fitness.

3.6 Visualisation of the Results

After the execution of the two algorithms, two Petri nets have been produced, and are reported in the following images.

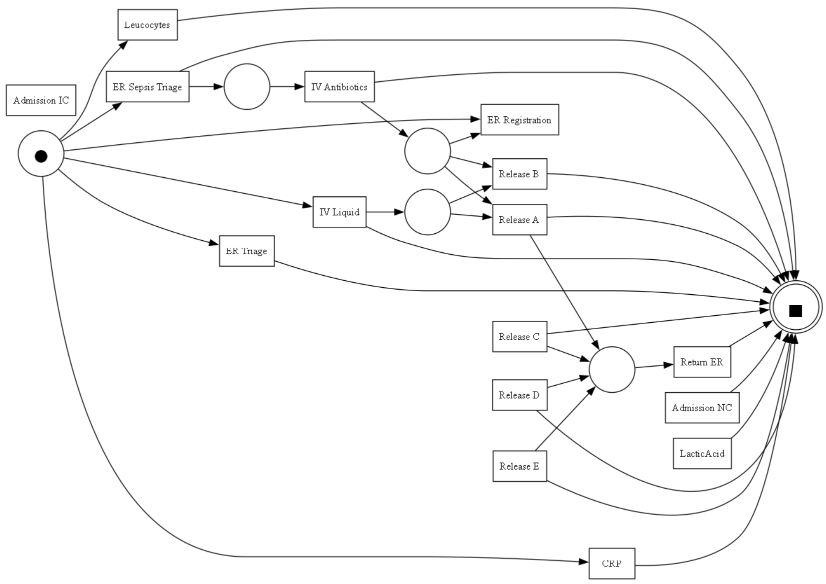

Figure 3.5: Petri net discovered using the Alpha Miner algorithm.

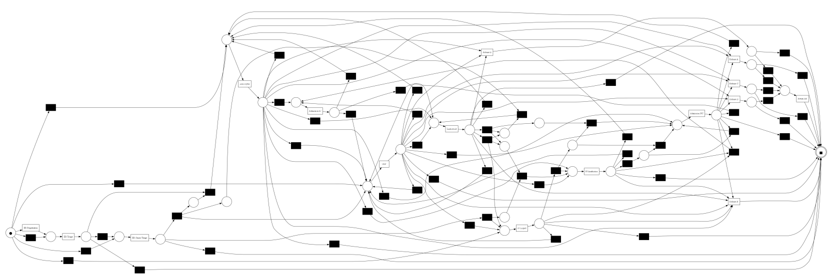

Figure 3.6: Petri net discovered using the Heuristic Miner algorithm.

4. Conclusion

This project explored the application of process mining techniques on a real-life event log concerning sepsis cases in a hospital. Using the *Disco* tool, an initial analysis was conducted to visualise the process flow, identify the most frequent paths, and examine the overall structure of the process. Disco proved to be an effective tool for exploratory analysis, providing intuitive visualisations and supporting interactive filtering and performance inspection.

Subsequently, two process discovery algorithms were applied using the `PM4Py` library to generate Petri nets from the same event log. The Heuristic Miner achieved a higher fitness score, demonstrating its ability to extract a more meaningful and generalisable process model.

This comparative analysis intended to explore the Disco tool and evaluate the differences between two algorithms studied during the Process Mining course. It highlighted the importance of choosing appropriate discovery algorithms depending on the nature of the data. While Disco is highly valuable for fast, user-friendly analysis, PM4Py provides fine-grained control for algorithm comparison and model evaluation. Together, these tools illustrate the complementary strengths of practical and academic approaches to Process Mining.

Bibliography

[1] Christian W. Günther and Wil M. P. Van Der Aalst. *Fuzzy Mining – Adaptive Process Simplification Based on Multi-perspective Metrics*. en. Ed. by David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Gustavo Alonso, Peter Dadam, and Michael Rosemann. Series Title: Lecture Notes in Computer Science. 2007. DOI: `10.1007/978-3-540-75183-0_24`. URL: `http://link.springer.com/10.1007/978-3-540-75183-0_24` (visited on 06/05/2025).

[2] *https://staff.icar.cnr.it/pontieri/didattica/PM/papers/fuzzyMining.pdf*. URL: `https://staff.icar.cnr.it/pontieri/didattica/PM/papers/fuzzyMining.pdf`.

[3] Felix Mannhardt. *Sepsis Cases - Event Log*. en. Dec. 2016. DOI: `10.4121/UUID:915D2BFB-7E84-49AD-A286-DC35F063A460`. URL: `https://data.4tu.nl/articles/_/12707639/1`.

[4] *pm4py: Process mining for Python*. URL: `https://processintelligence.solutions/`.

[5] *Process Mining and Automated Process Discovery Software for Professionals - Fluxicon Disco*. URL: `https://fluxicon.com/disco/`.

[6] *Process Mining with the Heuristics Miner-algorithm*. en. URL: `https://www.researchgate.net/publication/229124308_Process_Mining_with_the_Heuristics_Miner-algorithm`.

[7] *start — XES*. URL: `https://www.xes-standard.org/`.

[8] Eduardo Tiano. *edunando15/ProcessMiningReport*. June 2025. URL: `https://github.com/edunando15/ProcessMiningReport`.

[9] *What is Process Mining? — IBM*. en. Oct. 2021. URL: `https://www.ibm.com/think/topics/process-mining`.