Merging Directly-Follows Graphs and Sankey Diagrams for Visualizing Acyclic Processes

Ilya D. Derezovskiy, Nikita D. Shaimov, Irina A. Lomazova, Alexey A. Mitsyuk
HSE University, Faculty of Computer Science, PAIS Lab
Moscow, Russia
iderezovskiy@hse.ru, nshaimov@hse.ru, ilomazova@hse.ru, amitsyuk@hse.ru

***Abstract* — This paper proposes a method to visualize models of acyclic processes based on merging Directly-Follows Graphs (DFG) and Sankey diagrams. DFG is a popular graphical model to visualize discrete process models, while Sankey diagrams are used to represent flows of any kind. Our approach, based on flow diagrams, allows us to highlight individual cases or groups of cases in the overall model. The approach is implemented as a web-based tool that allows us, given an event log of an acyclic process, to construct and analyze the process behavior. We illustrate and evaluate the applicability of the proposed approach using learning processes as examples.**

***Keywords* — data visualization; directly-follows graphs; Sankey diagram; process mining; web application; data analysis**

I. Introduction

The visual representation of information makes it possible to analyze data quickly and in an accessible form. Visual analytics is widely used in the analysis of various processes [1], [2], [3], [4]. In the field of process analysis, process mining techniques are widely used to automatically synthesize and evaluate process models based on event logs [5], [6]. The resulting models are used to detect and eliminate flaws in existing processes.

Currently, process mining is particularly relevant and in demand. Recent research shows interest in the application of process mining in many fields such as business, healthcare and education [7], [8], [9]. Process mining techniques allow representing the behavior of a process in a visual form suitable for visual analytics [10].

Today, experts and researchers in the field of process analysis use various approaches to visualize process models. Specifically, processes can be represented as Directly-Follows Graphs (DFG), BPMN-models and Petri Nets. DFG is a simple and popular model that is easy to discover. However, due to its simplicity, it may represent the process too imprecisely. In particular, a DFG model can allow impossible behavior, which in turn can lead to incorrect conclusions about the features of the process [11]. An acyclic process has no cycles and can be considered as a flow of events. The Sankey diagram is often used to visualize various flows. This paper takes into account the advantages of both DFG and Sankey methods of visualizing processes and expands them to get a better visual representation of the acyclic processes.

The main goal of this paper is threefold: (1) to present a method to visualize an acyclic process model based on merging Directly-Follows Graphs (DFG) and Sankey diagrams that makes the analysis of such processes more convenient, (2) to illustrate and evaluate the proposed approach using real examples, (3) and to implement the proposed visualization method as a web application, describe its functionality and development technology.

The paper is organized as follows. Section II presents the problem we solve. Section III provides an overview of other existing solutions, highlighting their strengths and weaknesses. We discuss an illustrative motivating example in Section IV. Then, Section V presents the idea of our solution, whereas Section VI discusses its technical aspect. Section VII contains some findings and evaluations made when analyzing an example of visualization of a model of a real educational process, built from an event log. Finally, Section VIII concludes the paper.

II. Problem Statement

Digitized processes in various fields generate significant amounts of data, including event data reflecting user behavior. Through investigation of the process models, it is possible to obtain various insights about existing processes and come up with necessary changes in processes. The data does not take into account human relationships or other individual aspects. That means that decisions can not be performed in an automatic manner. Automated tools for process analysis should only help experts by providing the necessary information for decision-making.

It is a very important task to visualize generated models in an accessible and understandable way. There are many tools for visualizing process models, some of the popular ones are considered in the next section. However, existing tools do not provide an approach for the analysis of acyclic processes. Such processes occur when events in a process are arranged in a chain-like manner with no cyclic behavior. For example, an educational process where students attend lectures and seminars, perform homeworks and tests [7].

In this paper, we propose a way to visualize acyclic processes and present the tool that allows users to generate and visualize models of acyclic processes, highlight specified cases on the existing model and construct intersections for specified groups of cases.

We propose a way to visualize an acyclic process based on merging Directly-Follows Graphs (DFG) and Sankey diagrams. This method we implement as a client-server application that enables building models of acyclic processes from event logs, and visualizing them in the form of very specific oriented graphs. Moreover, the acyclic nature of the process makes it possible to compare different groups of cases within the process through the intersection of flows. The proposed visualization will simplify the analysis, search for relationships, deviations and anomalies in acyclic processes.

III. Overview of Existing Solutions

Many tools for visualizing process models based on event logs are available. We compare our tool with the following process mining tools and instruments for visualizing data in the form of various graphs and diagrams. Results of the comparison are summarized in Table 1 with the following numbering:

1) PMTK (novel web-based Process Mining ToolKit) [12].
2) Celonis (commercial online platform for analysis business processes) [13].
3) SankeyMATIC (website for building Sankey diagrams) [14].
4) Fluxicon Disco (application for analysis business processes) [15].
5) ProM (plugin-based framework for process mining) [16].
6) Proceset (analytical system for collecting and uploading data, conducting research in process mining) [17].
7) Solution proposed in this paper.

A. The Process Mining ToolKit (PMTK)

The Process Mining ToolKit, i.e., PMTK presents process mining algorithms and techniques in an easy-to-use solution. PMTK is built on top of the PM4Py library and allows non-technical users to use the advanced process mining technology implemented in PM4Py [18].

The important advantage is that PMTK provides a workspace in which the user is able to organize various event logs and projects. Subsequently, various objects, e.g., event logs and filters can be stored in the corresponding project's folder. Also, PMTK implements a process map with various filtering options (i.e., filtering of edges and activities). As such, PMTK, can be seen as a front-end solution for the advanced open-source process mining library PM4Py. But a user can not really interact with the graph: for example, one can not select vertices and transitions by clicking on the corresponding elements.

TABLE 1. Analysis of existing solutions

Feature	1	2	3	4	5	6	7
Function of automatic building models based on the event log	+	+	-	+	+	+	+
Function of visualizing models in the form of an oriented graph (diagram)	+	+	+	+	+	+	+
Interactive selection of vertices and transitions in the graph for filtering	+	+	-	+	+	+	+
Visualization of a model with intersection of subsets	-	-	-	-	-	-	+
Filtering by multiple values	+	+	-	+	+	+	+
Filtering by events and transitions	+	+	-	+	+	+	+
Creating and saving a subsets of cases	-	-	-	-	+	-	+
Saving all information about the user's projects	+	+	-	+	-	+	+
Free of charge for non-commercial and academic use	+	-	+	-	+	-	+

B. Celonis

Celonis is one of the leading products in the field of Process Mining. This commercial comprehensive software is designed to analyze, visualize and optimize processes within an organization based on data obtained from various IT systems. Including the analysis of educational processes.

The main advantage of Celonis is the ability to build different stages of processing applications on a straight line. By revealing different parts of the process, users can find out how many people participated in the processing of this incident, see the duration of each stage, and track statistics.

But it is difficult to integrate this online platform into the work of a non-commercial organization, since most of the functions are paid. There are also serious limitations in working with data: in particular, there are few data filtering options.

C. SankeyMATIC

SankeyMATIC is a free online tool that provides a wide range of controls that allow the user to customize the design of the constructed diagrams. User can export final design as either a PNG or SVG file. The advantage of this web tool is that it specializes in building Sankey diagrams.

However, SankeyMATIC does not provide the ability to create and manage projects, organize long-term work with data. There are serious limitations in working with data, as with previous analogues.

D. Fluxicon Disco

Disco's process development technology helps to build visual diagrams based on uploaded data. The program allows to perform process data analysis in order to optimize the efficiency of processes, control deviations, or explore various options for process tracks.

The advantages of the application are detailed statistics in a convenient visual form, convenient tools for filtering data, convenient tools for creating and working with user projects.

It is worth noting that a significant part of the functions is provided on a paid basis.

E. ProM

ProM is used to work with business process-related data and offers a variety of tools for analyzing, modeling, and improving business processes. ProM Tools provides a wide range of algorithms for process analysis, such as algorithms for detecting process models, algorithms for analyzing process performance, and many others. ProM also supports various process modeling standards, such as BPMN (Business Process Model and Notation), Petri Nets and others.

ProM is an extensible framework that supports a wide variety of process mining techniques in the form of plugins. It is platform independent as it is implemented in Java, and can be downloaded free of charge.

F. Proceset

Proceset is an active business intelligence system designed to manage and analyze business processes in real time, which allows you to quickly respond to changes and events within business processes.

The system integrates with various data sources, such as databases, process management systems (BPM), customer relationship management systems (CRM) and others, to obtain up-to-date information about processes.

Proceset provides visualization of business process data in a convenient and informative form, using only simple line graphs, process maps and tables.

An important disadvantage of the system, as with all the considered analogues, is also the inability to select subsets of cases using the specified filters from the general event log. It is also impossible to visualize models with multiple subsets with intersecting elements.

These analogues have a number of important common disadvantages:

– The inability to filter the event log (graph) by trajectory options through interactive selection of events and transitions (by clicking on the vertices and edges of the graph);

– The lack of visualization options for models with intersecting subsets;

– The inability to create and save subsets of cases.

IV. Motivating Example

Let us consider a process where in each trace events do not repeat themselves, and the resulting model is acyclic. It can be represented as a flow, where cases flow from one event to another. Generally, such cases are represented with DFG, where events and transitions from one event to another are shown as nodes and arcs whose thickness is based on the number of involved cases.

The problem with DFG is that in some cases it does not allow us to identify dependencies between events. As an example, consider the event log in Table 2, which contains data on students academic performance, and its DFG model in Fig. 1. For each trace in this event log, events «LR17 high» and «Accum low» do not occur in the same trace. However, the model in Fig. 1 allows a trace passing through these two events. Also, when we need to analyze certain groups of cases, the most common approach is to construct a separate model for this subgroup. Then an existing event log is filtered, leaving only the relevant cases, and a new model, based on the filtered event log, is synthesized and visualized. This leads to a quite cumbersome analysis and requires a visual comparison of two models.

To better reflect the flow of cases and dependencies between certain events, we can merge DFG with the Sankey diagram that is used to display flows. Such visualization allows to track changes in case flow at each moment of the process. As for analyzing groups of cases, a selected case or group of cases can be highlighted on the model as a separate flow. In this way, analysis can be performed on the same model without the need to create a separate model. Additionally, a selected group can be visually compared with the rest of the cases or with other groups.

TABLE 2. EXAMPLE OF AN EVENT LOG

Case_id	Activity	Timestamp	Grade	Lector_id	Group
#19273	LR1 high	08.11.2021	8	#13761	BPI203
#19273	LR2 med	09.11.2021	6	#13761	BPI203
#19273	LR3 med	16.11.2021	5	#13761	BPI203
…					
#45951	LR17 med	11.12.2021	6	#13976	BPI206
#45951	LR18 low	17.12.2021	3	#13976	BPI206

Let us take a look at an example with educational flow where students pass control elements of a course. For instance, consider the event log in Table 2. This log contains real data about students, grades received by students in a certain educational discipline, teachers and study groups. The grades for the test work are divided into pairs in accordance with the order in which they are carried out. For example, we want to take a look at students with low accumulated grade to find potential causes. Using the PM4PY python library, we can generate the DFG model of the process (see Fig. 1). Now we can apply filter to the event log and leave only traces containing «Accum low» events. Building a model with filtered log provides us with the following model (see Fig. 2). To search for potential causes both of the models should be analyzed side by side. The situation with the generation of Petri Net models is similar (see Fig. 3). Using DFG merged with the Sankey diagram, we can highlight traces with the «Accum low» event (see Fig. 4). Both models allow detecting differences between the specified group of students and the rest of the students. But DFG merged with the Sankey diagram eliminates the need to compare two models side by side and more clearly shows the flow of these cases in proportion to the others.

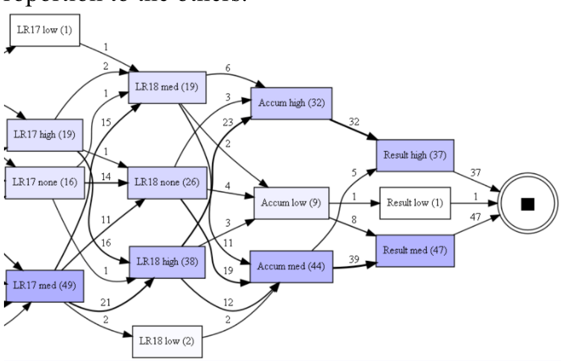

Fig. 1. An example of a DFG model of the educational process

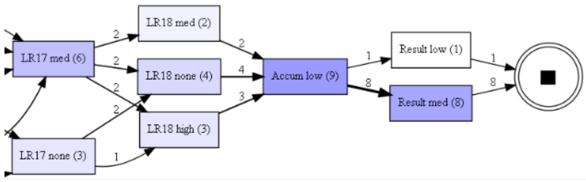

Fig. 2. The DFG model for traces containing the «Accum low» event

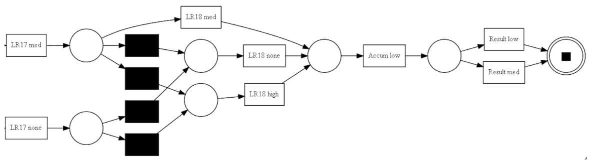

Fig. 3. The Petri Net model for traces containing the «Accum low» event

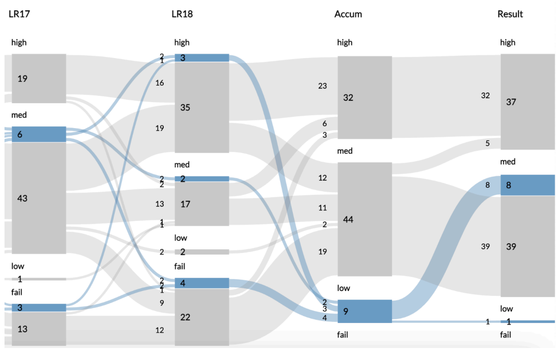

Fig. 4. The DFG model merged with the Sankey diagram with traces containing the «Accum low» event highlighted

V. PROPOSED SOLUTION

In this work, we propose the development of a tool in the form of a web application focused on the generation and visualization of acyclic process models. This tool will allow us to build interactive acyclic process models from event logs. The application provides the ability to specify subsets of cases from loaded event logs. The user can select subsets to be highlighted in the visualization, in order to display the trajectories of the selected subset among the other cases. In addition, it is possible to simultaneously visualize two selected subsets of cases and highlight the cases included in both subsets.

As a base for our data flow graphical representation, we chose a Sankey diagram. The classic version of the Sankey diagram, which is shown in Fig. 5, specializes on visualization of flows, but such diagrams do not contain vertices as such. This makes it difficult to view and analyze activities of a process, and, compared with the DFG, cuts off a significant part of the important information about the number of elements involved in the event.

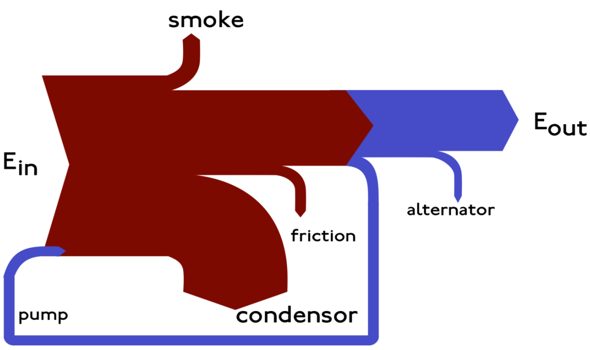

Fig. 5. An example of a classic Sankey diagram of a thermodynamic steam cycle [19]

The proposed solution to the problem is to merge the Sankey diagram with the DFG (see Fig. 6). By modifying the DFG, replacing the edges with wide transitions of the Sankey diagram and changing the size of the vertices in accordance with the number of cases involved, we obtain a visualization that clearly shows the proportional distribution and movement of cases throughout the process.

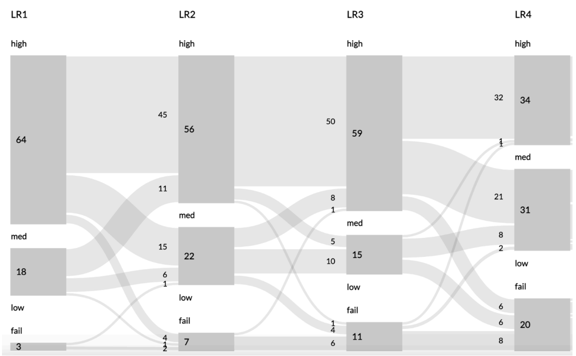

Fig. 6. An example of visualization for acyclic processes based on merging DFG and Sankey diagram

Let us review the obtained visualization for process models. The model columns represent process actions. For example, in the case of educational processes, they may correspond to academic disciplines or forms of control (seminar, test, exam, etc.). Columns may contain several nodes. Each node may represent a different action category in the process. For example, they can correspond to student grades (excellent, good, satisfactory, unsuccessful or «none» if the student missed a form of control). The arcs of the model show the transitions between events. Thickness of nodes and arcs correlate to the number of cases passing through them.

The resulting visualization is envisioned to be part of the web application. The application will allow us to bring interactivity into the visualization. For example, users can filter out a specific group of cases based on events and transitions through interaction with process visualization. This could be done by clicking on the corresponding elements (events or transitions).

Next we describe the main features of the application in more detail.

The application provides the ability to apply filters. Each applied filter is added to the list of currently applied filters. The selected set of cases is defined by the list of current filters. Cases can be filtered by the following categories:

1. *by events* – cases should or should not contain a specified event or events;
2. *by transitions* – cases should or should not contain a specified transition or transitions;
3. *by attributes* – cases should or should not contain certain attribute values. For example, for an educational process, an attribute can be specific student or teacher IDs, grades, etc.

In addition, we would like to emphasize the ability to create subsets of cases based on a list of filters. Cases that fit the current filters can be saved as a subset. The resulting subsets of cases can be highlighted in the model or used to create a visualization of the intersection of two selected subsets. Visualization of the intersection of two selected subsets shows one subset in one color and the other one in another color. Cases that present in both subsets will be shown as a gradient of two colors. The rest of the cases are shown in neutral color. As a result, visualization will display 4 different categories of cases that:

1. present only in the first subset;
2. present only in the second subset;
3. present in both subsets;
4. not present in any of the subsets.

It is important to mention the limitations of our visualization. The process must be acyclic, since cycles can disrupt the visual coherence of the model. Therefore, each action for each case in the event log should occur no more than one time.

The application provides an account system to implement the saving of the user's work progress. After registering or logging in, the user can create a project and upload the necessary event logs. Projects store uploaded event logs and created subsets of cases.

And of course the most important function of the application is model generation using uploaded event logs and subsets. Generation takes place at the time of request for visualization of the process model and mostly follows standard DFG generation algorithm. Unique event names in the event log create the set of graph vertices and edges reflect direct transitions from one event to another.

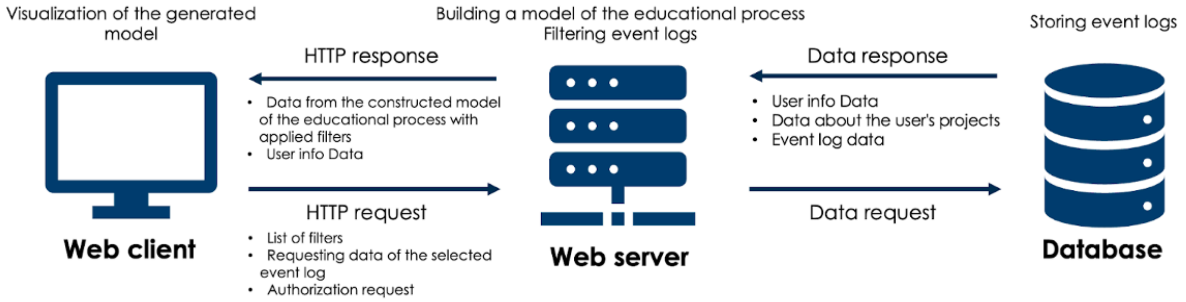

Fig. 7. The software architecture

VI. Tool Architecture and Implementation

Let us describe the architecture of the developed web-based application. It follows the client-server application model and shown in Fig. 7. The main three components of our client-server application are as follows:

· **Web client** handles and displays the visualization of the generated model. It communicates with the server via HTTP requests, providing information about the selected event log data, applied filters, and authorization requests;

· **Web server** constructs educational process models and filters event logs. Upon receiving requests from the web client, it processes the data and returns an HTTP response containing the educational process model with applied filters, along with user data;

· **Database** stores event logs and responds to data requests from the server. It provides information about users, projects, and event logs back to the server when it receives a request.

The user interface (the client-side part of the application) is implemented in JavaScript using HTML and CSS. This combination consists of the typical technologies used to build web-based applications. JavaScript provides dynamic behavior and interactivity, while HTML defines the structure of web pages, and CSS handles presentation and styling. It is supposed to use a JavaScript library React JS for creating user interfaces. This library supports component-based architecture, which makes it easier to manage and reuse UI elements. React's document object model helps update the user interface efficiently, leading to faster rendering and an improved user experience. This architecture encourages cleaner code organization via components, promoting improved maintainability and scalability of web-based applications. The part of the program responsible for plotting and visualizing graphs is implemented using the open-source JavaScript library D3.js (Data Driven Documents 7.6). This library provides extensive capabilities for creating custom visualizations, allowing developers to design bespoke charts and graphs tailored to specific needs. D3.js uses SVG, HTML and CSS to render visualizations. This provides flexibility and control over the appearance and behaviour of the elements, giving more power over data representations.

The developed tool for visualizing acyclic processes is implemented as a client-server application. Python was chosen to implement the backend of the application. It should be noted that a lot of Python frameworks are available to implement the backend of the web application, including Django, Flask and Pyramid. These frameworks provide opportunities for developing server-side components of applications, such as handling HTTP requests and communicating with the backend of the application with databases. An important advantage of using Python for programming is that it can be easily integrated with other programming languages. It is especially important that it can be used in combination with JavaScript. Additionally, Python is widely used for data analysis, including in the field of process mining. Python provides many libraries and frameworks for analyzing and visualizing process data: numpy, pandas, pm4py, and others.

In our application, PostgreSQL has been chosen as the database management system for storing information about users, created projects, and uploaded event logs. In general, PostgreSQL supports the management of databases of unlimited size.

Visual Studio Code is a popular web application development tool which supports programming languages such as JavaScript, HTML, and CSS, as well as Python, React JS, which we used for the development of our visualization tool.

VII. Findings and Evaluations

Let us now provide some observations that were made in the course of analyzing the visualization of the model built on the basis of event logs of the real learning process. The event logs contain information about students' grades for the algebra practicum course.

Fig. 8 shows the visualization with the intersection of two subsets of cases. The first subset contains students who received an excellent grade for the last work (node «LR18»), and the second subset contains students who received a low accumulated grade (node «Accum»). The resulting grade for the discipline is formed as follows: the resulting grade consists of the accumulated grade and the final exam grade, the resulting grade is a sum of those grades where each grade has a certain coefficient. The accumulated grade

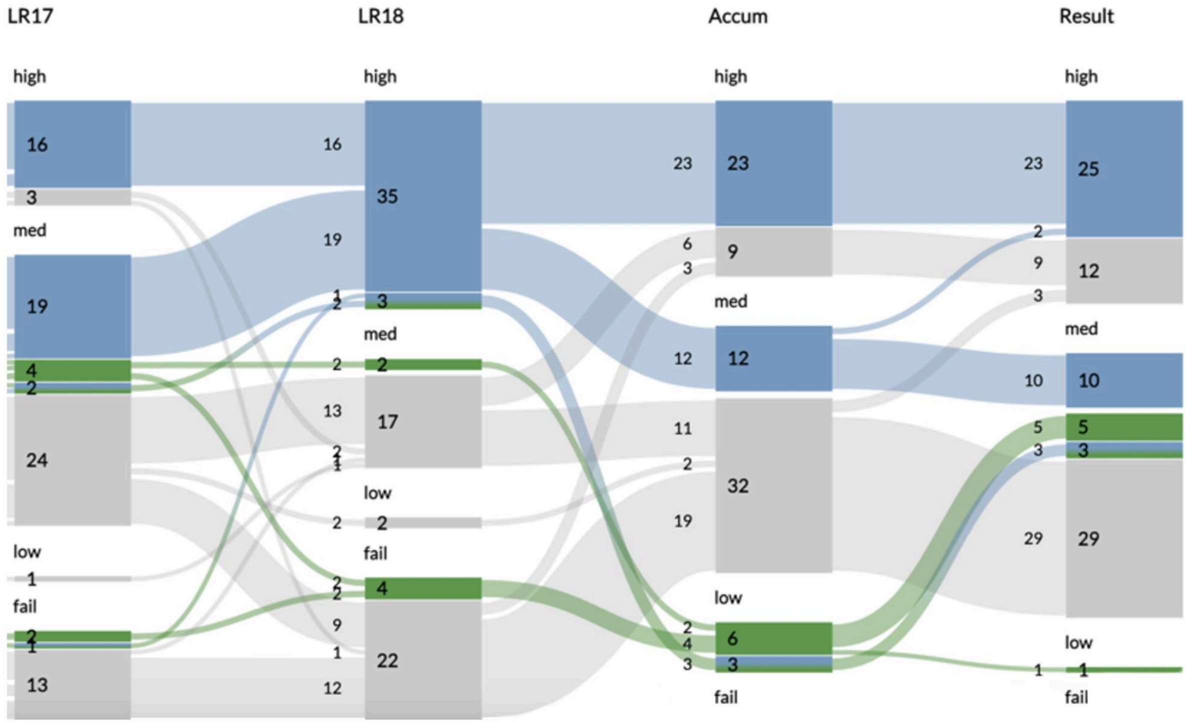

Fig. 8. Visualization of the intersection of two subsets of cases

is a sum of all the grades that the student received throughout his studies in the discipline, each with their own coefficient. The final exam takes place at the end of the academic discipline. The visualization shows the trajectories of students from the two subsets, and students who belong to the intersection of these subsets (those who had received an excellent grade for the last test work and a low accumulated grade), are highlighted by a gradient.

From the model, it is clear that most of the students who received a low accumulated grade received a medium final grade. This may indicate that these students received a medium or high mark for the final exam and successfully completed the course. To look at the trajectory for such students throughout the course, we can click on the transition between the «Accum low» and «Result med» vertices to form a subset of students who contain this transition in their trajectory. The subset, highlighted in blue color, is displayed on the model (see Fig. 9).

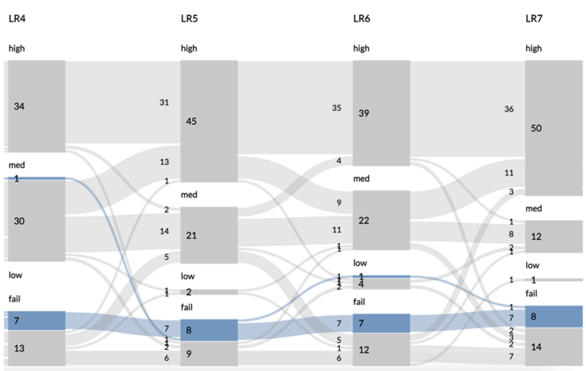

Fig. 9. Visualization of the subset of students

Upon further analysis of the diagram, it becomes clear that these students missed a large number of assessments, resulting in a low accumulated grade.

Detailed analysis of the educational process models combined with visualization for subsets of students will help to identify areas and events that may be a cause for concern and require special attention from educational program administrators.

VIII. Conclusion

In this paper, we presented a new method for visualizing acyclic processes. The visualization is based on merging of DFG and Sankey diagrams, taking advantage of the benefits of each. Our approach, based on the flow diagrams, allows us to highlight individual cases or groups of cases in an overall model. The approach is implemented as a web-based tool that allows us, given an event log of an acyclic process, to construct and analyze the process behavior. It is assumed that in combination with other data analysis methods, the causes of particular events and trajectories can be easily identified.

The application provides the ability to specify subsets of cases from uploaded event logs to highlight them in the visualization. In addition, one can simultaneously visualize two selected subsets of cases and highlight overlapping cases that are included in both subsets.

As an example of the practical usage of the developed application, we show how it allows analysts and managers of educational programs to analyze data about the learning process and students' trajectories by visualizing educational trajectories in order to identify deviations, relationships and anomalies that require special attention from administrators of educational programs.

An important constraint of our solution is that it deals with acyclic models only. For future work we plan to generalize our visualization method for models with cycles.

ACKNOWLEDGEMENT

Support from the Basic Research Program of the National Research University Higher School of Economics is gratefully acknowledged.

REFERENCES

[1] T. Gschwandtner, «Visual Analytics Meets Process Mining: Challenges and Opportunities», in Data-Driven Process Discovery and Analysis. SIMPDA 2015. Lecture Notes in Business Information Processing, vol 244. Springer, Cham, 2017, pp. 142–151. doi: 10.1007/978-3-319-53435-0_7.

[2] Y. V. Kotylev and A. A. Mitsyuk, «Software System Behavior Can Be Analyzed with Visual Analytics», in Proceedings of the Conference on Modeling and Analysis of Complex Systems and Processes 2020 (MACSPro 2020) / Ed. by Alexander Shapoval, V. Popov, I. Makarov. Vol. 2795. CEUR Workshop Proceedings, 2020. pp. 46-56.

[3] S. Miksch, «Visual Analytics Meets Process Mining: Challenges and Opportunities», 3rd International Conference on Process Mining (ICPM 2021), Eindhoven, Netherlands, 2021, pp. xiv-xiv, doi: 10.1109/ICPM53251.2021.9576854.

[4] J. Rehse, L. Pufahl, M. Grohs and L. Klein, «Process mining meets visual analytics: the case of conformance checking». arXiv preprint arXiv:2209.09712 (2022). doi: 10.48550/arXiv.2209.09712.

[5] W. M. P. van der Aalst, Process Mining - Data Science in Action, Second Edition. Springer, 2016, isbn: 978-3-662-49850-7. doi: 10.1007/978-3-662-49851-4. [Online]. Available: https://doi.org/10.1007/978- 3--49851-4.

[6] J. Carmona, W. M. P. van der Aalst. Process Mining Handbook: Lecture Notes. Springer Nature, Switzerland, 2022. 504 p. doi: 10.1007/978-3-031-08848-3.

[7] N. D. Shaimov, I. A. Lomazova, A. A. Mitsyuk, and I. Y. Samonenko, «Analysis of Students' Academic Performance using LMS Event Logs», Modeling and analysis of information systems, vol. 29, no. 4, pp. 286-314, 2022.

[8] J. Munoz-Gama, N. Martin, C. Fernandez-Llatas, O. Johnson, M. Sepúlveda, E. Helm, et al. - Process mining for healthcare: Characteristics and challenges. J. Biomed, 2022. doi: https://doi.org/10.1016/j.jbi.2022.103994.

[9] P. Badakhshan, B. Wurm, T. Grisold, J. Geyer-Klingeberg, J. Mendling, J. Vom Brocke - Creating business value with process mining. J Strateg Inf Syst., vol. 31(4), 2022. doi: https://doi.org/10.1016/j.jsis.2022.101745.

[10] I. Sitova and J. Pecerska, «Process Data Analysis Using Visual Analytics and Process Mining Techniques», 61st International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS), 2020. doi: https://doi.org/10.1109/ITMS51158.2020.9259296.

[11] Wil M.P. van der Aalst, «A practitioner's guide to process mining: Limitations of the directly-follows graph», Procedia Comput. Sci., vol. 164, pp. 321-328, 2019. doi: https://doi.org/10.1016/j.procs.2019.12.189.

[12] The Process Mining ToolKit (PMTK) Documentation. [Online] Available: https://pmtk.fit.fraunhofer.de/ [Accessed Mar. 05, 2024].

[13] The Celonis Documentation. [Online] Available: http://celonis.com/ [Accessed Mar. 05, 2024].

[14] The SankeyMATIC Documentation. [Online] Available: https://sankeymatic.com/ [Accessed Mar. 05, 2024].

[15] The Fluxicon Disco Documentation. [Online] Available: https://fluxicon.com/disco/ [Accessed Mar. 05, 2024].

[16] The PromT Documentation. [Online] Available: https://promtools.org/prom-documentation/ [Accessed Mar. 05, 2024].

[17] The Proceset Documentation. [Online] Available: https://infomaximum.ru/docs?v=02.2024 [Accessed Mar. 05, 2024].

[18] Alessandro Berti, Sebastiaan van Zelst, Daniel Schuster, «PM4Py: A process mining library for Python», Software Impacts, vol. 17, p. 100556, 2023. doi: 10.1016/j.simpa.2023.100556. [Online]. Available: https://doi.org/10.1016/j.simpa.2023.100556 [Accessed Mar. 25, 2024].

[19] Wikimedia Commons, the free media repository. [Online]. Available: https://www.google.com/url?q=https://commons.wikimedia.org/wiki/File:Sankeysteam.png&sa=D&source=docs&ust=1712231868106450&usg=AOvVaw389upF5qS7BipkMnvrzqon [Accessed Mar. 30, 2024].