ESub: Exploration of Subgraphs
A tool for exploring models generated by Graph Mining algorithms

Claudia Diamantini, Laura Genga, and Domenico Potena

Information Engineering Department
Università Politecnica delle Marche
via Brecce Bianche, 60131 Ancona, Italy
{c.diamantini,l.genga,d.potena}@univpm.it

Abstract. In this demo we introduce *ESub*, a tool aimed at visualizing the outcome provided by a frequent subgraph mining algorithm, i.e. SUBDUE. Such a tool has been developed as a supporting tool for a methodology we proposed in previous works for analyzing unstructured processes, based on the use of graphs. By exploiting graphs-based techniques, it is possible to provide the user with a different perspective on a process, where only the most relevant subprocesses (i.e., subgraphs) are displayed, rather than the complete, end-to-end process schema, which often results very chaotic in unstructured domains. Our tool allows the user to visualize and interact with such subgraphs. Furthermore, it allows for visualizing the original graphs of the set, and compress them by means of the most relevant subgraphs, in order to obtain a simplified view of the overall process.

1 SubProcesses Analysis

Process Mining (PM) methods are aimed at extracting from an event log a process schema describing the flow of the performed activities [1]. However, when applied to the so-called "Spaghetti Processes", i.e. processes with little or no structure, classical PM techniques usually generate a very chaotic model, usually not understandable for a human analyst. As a remedy, in previous works [2] we proposed an alternative methodology for analyzing spaghetti processes, aimed at extracting their most relevant subprocesses. Such approach exploits a graph-based technique; more precisely, it requires to transform the event log of the process in a set of graphs, each of them describing the execution of a certain process instance, then extracting the subprocesses from these graphs.

In this paper, we present *ESub*, a tool we developed to manage the set of subprocesses extracted from a spaghetti process, represented as subgraphs. ESub is designed to offer advanced functionalities for the subprocesses visualization and analysys. Furthermore, it also provides the user with a flexible mechanism to simplify the overall process visualization by exploiting a compression mechanism, i.e. by replacing each subgraph with single nodes. The user can set the desired level of compression.

The current implementation of our tool is aimed to manage the set of subgraphs extracted by the SUBDUE algorithm [4], that is a hierarchical clustering algorithm. Anyway, it is easy to build adapters to apply our tool also to results obtained from other FSM algorithm. The outcome provided by SUBDUE is a hierarchy, where the top level subgraphs are built by using only elements of the original graphs set (i.e., nodes and edges), while lower level subgraphs involve the higher level subgraphs in their definition. Typically, the top-level subgraphs represent the most relevant ones. SUBDUE also labels each subgraph on the basis of the order in which they have been extracted. We discussed in [3] a set of measures to evaluate the SUBDUE subgraphs; currently, two of them are taken into account by ESub, i.e. the *frequency* (FREQ), which evaluates the number of occurrences of a subgraph in a graphs set, and the *representativeness* (REP), which evaluates the percentage of graphs in which a subgraph occurred at least once.

Our tool is implemented as a web application. The most of the available functionalities can be called from a left side menu, organized in a set of tabs, as shown in Figure 1; the user can expand each tab by clicking on its label. The following subsections describe the two main groups of functionalities offered by the tool, i.e. the *visualization* and the *compression* of subgraphs.

1.1 SubProcesses Visualization

There are two main kinds of visualization functionalities, i.e. the a)*navigation*, and the b)*filtering* of subgraphs. As regards group a), first the user has to use the *load* tab to upload the file of the subgraphs to visualize. From the load tab, the user can chose if upload the outcome returned by SUBDUE, that is a SUBS file, or a more generic DOT file, depending on which outcome the user has at disposal. It is also possible, although not mandatory, to load a LOG file, that stores the FREQ and REP values for the subgraphs, which can be exploited during the subgraphs exploration.

After the upload, each subgraph is displayed as a single node, with the label assigned to it by SUBDUE. Such compact representation provides an overview of the complete hierarchy extracted by SUBDUE. The user can, anyway, expand a node by selecting it and then using the *expand* tab, or by simply double-clicking the node. Similarly, it is possible to compress an expanded node by selecting it and using the *compress* tab or by double-clicking the node. The expand/compress tabs allow the user also to expand/compress several nodes (possibly, the entire hierarchy) at the same time. Figure 1 shows the uploaded subgraphs set, with the nodes SUB_2 and SUB_{31} expanded. Note that SUB_2 is a “parent" of SUB_{31}, since SUB_{31} involves SUB_2, as we can see in the figure. Using the tab *Layout*, it is possible to adjust the visualization according to the user's needs; in particular, it is possible to increase/decrease both the width and the height of the displayed outcome. It is also possible to enable/disable the movement of the nodes. Note that there is also a right-side menu, which displays the list of the subgraphs that are currently compressed/expanded; by clicking the name of a compressed/expanded subgraph, the corresponding node in the hierarchy is surrounded with a red square.

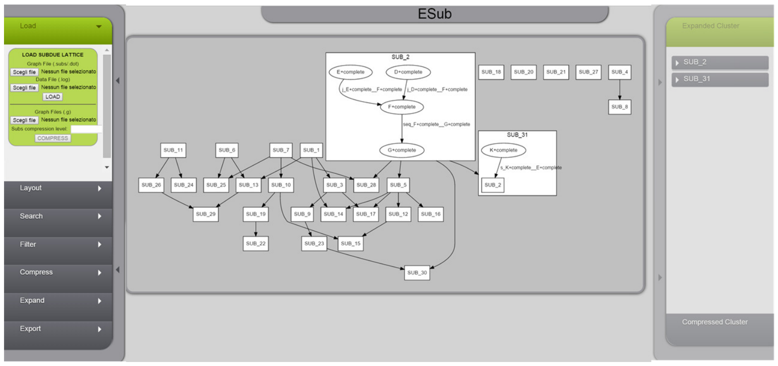
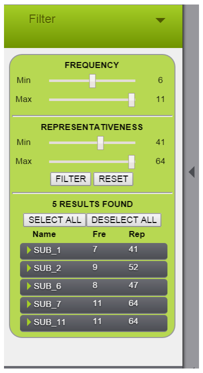


Fig. 1: Uploaded subgraphs set, with two subgraphs expanded

Functionalities of group b) aim at supporting the users in easily detecting the most interesting subgraphs. It is unrealistic that the user can analyze each single subgraph, since they can be hundreds. Hence, the tool implements some functionalities aimed to support the search, or the filtering, of specific subgraphs. The first, and the most simple one, is the *search* tab, which allows the user to search the subgraphs by using user's keywords. The other search functionality is provided by the *filter* tab, which allows for detecting the subgraphs which correspond to certain values of FREQ and/or REP. It is also possible to use the filter functionality without changing the default parameters setting; in this case, it will list all the top level subgraphs, reporting their values of FREQ and REP. It is interesting to note that the filtering functionality takes into account only the top-level subgraphs; in fact, since they are assumed to be the most relevant ones, the FREQ and REP values are computed only for them.

We would like to point out that the tool also allows the user to *export* either the entire hierarchy or a set of selected subgraphs. More precisely, by using the export functionality it is possible to export the visualized hierarchy either in a SVG or in a DOT format; the corresponding file is automatically downloaded. Furthermore, it is also possible to select a set of subgraphs and visualize them in another web page; this is extremely useful when only a subset of the hierarchy has to be analyzed. Note that in the new page, by selecting again the export tab, the tool generates the SVG or DOT file corresponding only to the exported portion of the hierarchy.



FREQUENCY

Min 6

Max 11

REPRESENTATIVENESS

Min 41

Max 64

5 RESULTS FOUND

Name	Fre	Rep
SUB_1	7	41
SUB_2	9	52
SUB_6	8	47
SUB_7	11	64
SUB_11	11	64

Fig. 2: Filtering Tab

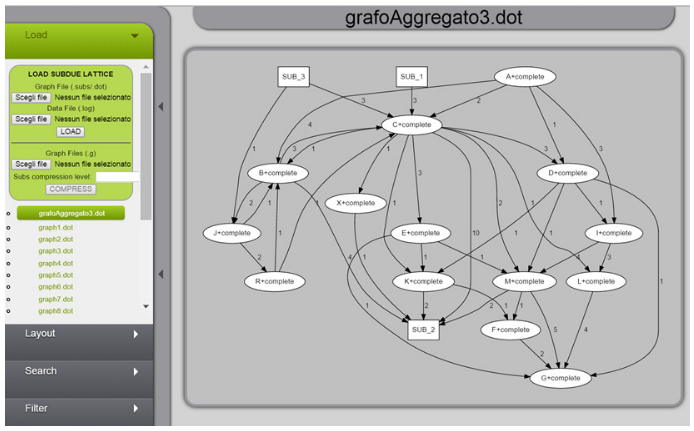
1.2 Compression

The goal of these functionalities consists in obtaining a simplified model both of each single process instance and of the overall process, obtained by aggregating the instances. Such a simplification is performed by applying a compression mechanism, which consists in replacing each extracted subgraph with a single node. The user can set the desired level of compression.

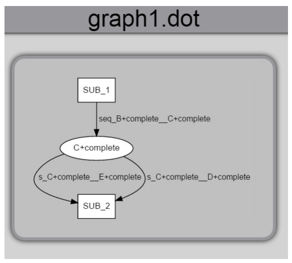
The user has to upload the file SUBS again, but this time has to use the second component of the *load* tab, which allows her to upload the graphs she intends to use and to select the level of compression she wants to achieve. After the uploading, ESub displays the aggregated graph. Figure 3a shows the outcomes of the compression where 17 graphs (each representing a specific instance of a process) have been uploaded, with the same file SUBS we used before. For this example, we set the level of the compression to 3. Note that selecting a level compression means indicating which subgraphs one wants to use to compress: for example, by selecting 3 we are indicating that we want to use SUB_1, SUB_2, SUB_3. In the graph, the nodes representing the replaced subgraphs have a different shapes (square), and are labeled with SUB_n: by double-clicking them, another page is opened, which visualizes the subgraph. The user can also select a single compressed graph from the list of the tab, to explore the compression of a single graph; for example, Fig 3b shows the first compressed graph. It is interesting to note that the user can also visualize the original graphs set and, hence, the not-compressed aggregated graph, by simply setting a level of compression equal to zero during the files uploading.

2 Screencast

A screencast showing an example of use of the GraphManager tool can be found at `http://kdmg.dii.univpm.it/?q=content/subprocess-discovery`, together with the files we used during the examples discussed in this paper.



(a)



(b)

Fig. 3: The Aggregated compressed graph (a) and a compressed graph (b)

3 Link

The ESub tool can be found at `http://kdmg.dii.univpm.it/?q=content/subprocess-discovery`.

References

1. Van der Aalst, W.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer (2011)
2. Diamantini, C., Genga, L., Potena D., Storti, E.: Patterns discovery from innovation processes. In: Proc. of the 2013th International Conference of Collaboration Technologies and Systems, pp. 457-464, San Diego, USA, May 20-24 2013. IEEE (2013)
3. C. Diamantini and D. Potena. Hierarchical clustering of process schemas. In: Proc. of the 3rd Interop-Vlab.It Workshop on Enterprise Interoperability, pp 27–32, Naples, Italy, Oct. 9 2010. (2010)
4. I. Jonyer, D.J. Cook, and L.B. Holder. Graph-based hierarchical conceptual clustering. J. Mach. Learn. Res. 2. (2002) 19–43