Project Initiation

Label Refinement by Behavioral Similarity

Document owner:
Bianka Bakullari
Christopher Beine
Nicole Ventsch
Juan

Last edited: April 19, 2019

1 Overview

Many processes involve carrying out an action multiple times. An example for this would be an online shop in which you first have to pay a registration fee before ordering an item and paying it. This process contains the event "payment" twice, but in different contexts, so that the payments are actually two different tasks. In the context of analysing processes, the event logs usually only contain the event names, so that the "payment" actions would be treated as the same task and loops would be induced in the resulting models. However, these loops do not match the actual process, which is the issue this project addresses.
These imprecise logs should be refined based on the structural contexts of the events. We want to refine the logs without any filtering. Moreover, we want to allow an interactive change of the thresholds used to refine the labels since this can differ for every log and we have no knowledge of the correctness of the refined log in general. All of this should be done by creating a Python-based web service in this project.

2 Business Case

Based on the situation explained in the overview, we now want to discuss the business case and explore the project's potential. We will do this by first defining the scope of the project, then the assumptions we take and finally we will consider the advantages that are gained by carrying out this project.

2.1 Scope

During the project, we will create both code and documentations. Thus, the scope will be divided into these two aspects:
Documentation:

- develop and describe the design of the interface we will use and picture how users can set the thresholds in the interface
- design the algorithm structure by stating the usage of classes as well as the inputs and outputs for the functions that are used

Implementation:

- set up a Web Service based on Python that uses the label refinement algorithm proposed by Xixi Lu, et al.
- create a user interface that allows the users to upload the original event log, set thresholds and imprecise label scope and finally download the refined event log

Moreover, we will follow the software development Lifecycle and, in this connection, we will create documentation on the project initialization, on the requirement specifications, on the design and P.o.C and a software documentation. Each of them have to be submitted before the corresponding due date stated in the section "Project Plan - Milestones".

2.2 Assumptions

In this project we will assume that an event log is given by the user, i.e., data that contains at least the attributes "id", "time stamp" and "activity name". Moreover, we will assume that these event logs are given in the standard XES format.

2.3 Key Benefits

With this project we mainly aim at improving given event logs by making them more precise, so that the subsequent analysis will be more accurate. Using this approach, the process logs can be refined to reach a higher precision in the subsequent analysis of up to 89 %, which highly increases the quality of the process discovery results. Thus, the data scientists will be enabled to gain better insights from the data. These insights can then lead to more optimized processes and that way reduce the company's expenses while increasing its efficiency.
By designing an interface that allows the users to upload an event log, to set the thresholds and to download the modified event log, carrying out this project will save the data analysts a lot of time which would be

needed to refine the log themselves. By providing a web service, the software can be used on demand and there are only minor requirements for the user's hardware and avoids the necessity to stick to a given programming language or software.

3 Feasibility study

3.1 Theoretical point of view

From a theoretical point of view, we view an event log for which we suspect to have different tasks being handled as the same activity as *imprecise*. In reality, each event in the data is a unique occurrence over time. By defining a *labeling function* which maps these events to a finite set of activity names, also called *labels*, we obtain event logs having common activities across traces and within traces. A problem arises when similar events happening in different contexts are assigned the same label. Applying process discovery algorithms to such event logs may lead to process models being imprecise, misleading or even incorrect.
There has already been some research investigating this problem. Some of the approaches include using additional domain knowledge to correct the labeling, or trace clustering to do relabeling according to the different variants.
In our project, we focus on the algorithm developed, investigated and explained in [1], which given an event log, refines the labels of events based only on the context and patterns present in this event log. The approach consists of three main steps: 1)detecting activities which need refined labels, do relabeling 2)accross dissimilar traces and then 3)within traces. In our project, we assume that the set of activities needing relabeling is given and the user should have the possibility to set the thresholds affecting the relabeling in steps 2) and 3).
Since the logs have a finite set of activities leaving room for only a reduced number of computations needed for the cost function in step 2) and the fact that this algorithm has already been successfully implemented in ProM yielding good results, we conclude that the project is feasible from a theoretical point of view.

3.2 Technical point of view

As stated in [1], the algorithm has already been implemented in ProM, where controlled experiments have been made to test the performance of the algorithm based on event logs containing different patterns. There has also been a real life case study involving an event log with data concerning healthcare which was provided by Maastricht University Medical Center (MUMC+), a large academic hospital in the Netherlands. In the log containing 1039 cases and 6213 events, the activities *surgery* and *consultation* were imprecise. After relabeling, the resulting model reflected the behaviour which was described by the domain experts, according to whom the acitivity *surgery* for example took place many times during the treatment of certain patients.
For our project we will use Python as back-end programming language to implement the algorithm.
@all front end, libraries used and why
********** under construction **********
@all Do we need external libraries to implement the algorithm?
TODO: formulate bullet points [to be done until friday]

- Flask 1.0.2 as web application framework
- Python 3.7.2 as back-end programming language
- pm4py python library as process mining toolkit
- NetworkX pyhton library for graph manipulation
- JavaScript as Front-end development language (Standard ECMAScript 2018)
- Browser support (Google Chrome version $\geq$ 70, Firefox version $\geq$ 63, Maybe don't support IE/Edge)
- TODO: Select HTML5 visualisation framework for petri nets [To be done until saturday]
- TODO: Select JavaScript Framework / Actually Required? [To be done until saturday]
- Bootstrap version 4.3.1 for fast ui design

flask

- References: Netflix Lemur, Linked-in internal stack
- Minimalistic Framework
- highly customizable

Python

- Specified by stackholder
- highly used for datascience
- reference

JavaScript

- defacto standard
- Required for dynamic content

Bootstrap

- References: Twitter, Spotify, Coursera
- Fast Responsive UI design (May indicate responsive as optional for project)

****

3.3 Risks and migiations

3.3.1 Project management risks

Risks	Mitigations
Misunderstanding of Tasks	Do regular discussiond, meetings in the group and with the project supervisor
Unmatching schedules of the team members	Arrange meetings in advance, notify other members for any change
Acquiring new skills under time pressure	Assign tasks based on individual strengths to increase efficiency

3.3.2 Technical risks

Risks	Mitigations
Inconsistencies in programming styles	Explain the code to other members, write useful comments while coding
Unclear performance measures	Test code on small event logs, use automatically generated event logs to experiment
Software components do not work as a whole	Put emphasis on the design step

4 Project Plan

4.1 Milestones

The project starts on the 09/04/2019 and ends on the 08/07/2019 and is divided into nine milestones. The project is managed with a scrum-oriented approach where each milestone represents a sprint. The project team organizes the required tasks during each sprint and visualize the current project status via a dashboard.
TODO: add sprint description until friday

Table 1: Overview Milestones

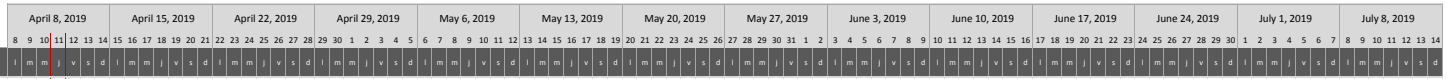
ID	Milestone	Description	Deadline
1	Project Initiation document	The Project Initiation Document provides all of the key information required to start and run the project. This includes the project description, business case, feasibility study and a project team presentation.	19/04/2019
2	Requirements Specification document	The Requirements Specification document contains functional and none functional requirements such as a set of use cases to describe the system interactions.	29/04/2019
3	Design Analysis and dummy P.o.C.	The final document is a description about the planned system architectural background and a proof of concept visualizing the main UI components.	13/05/2019
4	Sprint 1 code and documentation	TODO: sprint description depending on GANTT chart	24/05/2019
5	Sprint 2 code and documentation	TODO: sprint description depending on GANTT chart	07/06/2019
6	Sprint 3 code and documentation	TODO: sprint description depending on GANTT chart	21/06/2019
7	Testing, assessment and deployment	The application is checked for accuracy and should be aviable for use.	01/07/2019
8	Final report on the project	The final report provides an overview about the project course and the result.	08/07/2019

Label Refinement based on Behavioral Similarity

Project Start: 11/04/19

Display Week: 1

TASK	START	END	DAYS
Project Initiation			
Set up the Project Office	11/04/19	12/04/19	2
Develop a Business Case	11/04/19	16/04/19	6
Undertake a Feasibility Study	12/04/19	17/04/19	6
Establish the Project Charter	15/04/19	19/04/19	5
Appoint the Project Team	16/04/19	21/04/19	6
PI document elaboration	11/04/19	21/04/19	11
Requirements Specification			
Requirements analysis	22/04/19	24/04/19	3
Use case analysis	24/04/19	26/04/19	3
Requirements review	26/04/19	28/04/19	3
RS document elaboration	22/04/19	28/04/19	7
Design Analysis & dummy P.o.C.			
Architectural design	29/04/19	01/05/19	3
High-level design	01/05/19	03/05/19	3
Detailed design	04/05/19	12/05/19	9
Frontend mockup	05/05/19	09/05/19	5
Documentation review	29/04/19	12/05/19	14
Sprint 1			
Web Service set up	13/05/19	15/05/19	3
Static frontend	13/05/19	17/05/19	5
File loader module	13/05/19	17/05/19	5
Unit test	17/05/19	18/05/19	2
Module integration I	18/05/19	24/05/19	7
Documentation review	13/05/19	24/05/19	12
Sprint 2			
Imprecise label detetion module	24/05/19	29/05/19	6
Cost function module	24/05/19	31/05/19	8
Mapping module	24/05/19	31/05/19	8
Unit test	01/06/19	03/06/19	3
Module integration II	04/06/19	07/06/19	4
Documentation review	24/05/19	07/06/19	15
Sprint 3			
Horizontal/Vertical refinement module	10/06/19	17/06/19	8
File exporter module	10/06/19	13/06/19	4
Petrinet visualization module	10/06/19	14/06/19	5
Unit test	15/06/19	17/06/19	3
Module integration III	17/06/19	20/06/19	4
Documentation review	10/06/19	21/06/19	12
Testing, assessment and deployment			
Integration testing	22/06/19	25/06/19	4
Internal project assessment	26/06/19	27/06/19	2
Deployment	28/06/19	29/06/19	2
Final documentation	22/06/19	01/07/19	10



4.2 Deliverables

With each milestones various deliverables are created to monitor, document and verify the docuemnt progress.

1 Project Initiation document:

- Finale project initiation document with key information about the project

2 Requirements Specification document:

- Requiredments Specification document with functional und none functional requirements
- Use case analysis

3 Design analysis and dummy P.o.C:

- System and software architecture documentation
- Frontend mockup

4 Sprint 1 code and documentation

- Python components
- Unit Test protocols
- Code documentation

5 Sprint 2 code and documentation

- Python components
- Unit Test protocols
- Code documentation

6 Sprint 3 code and documentation

- Python components
- Java Script components
- Unit Test protocols
- Code documentation

@all: should we write tests during the implementation? Could possibly save us a lot of trouble, but we have to select a test framework. Otherwise we can remove the Unit Test protocols

7 Testing, assessment and deployment

- Testprotocols
- Server configuration
- Web API
- Webapplication

8 Final report on the project

- Final report

@all: Any other documents, reports, protocols or code artefacts required?

4.3 Timetable

5 Project Team

5.1 Competences

5.1.1 Nicole Ventsch

Nicole Ventsch is a Master student at RWTH studying Mathematics and Data Science in parallel. She is studying both subjects in her second semester of the master's degree. Moreover, she works as a student assistant in the field of Data Analytics / Business Intelligence. Due to her background in mathematics, she has a good understanding of theoretical foundations. Moreover, she is very interested in Data Science and already took many courses in that area. Since she took the course "Introduction to Data Science", she also worked with Python before. Though she has a strong theoretical background, she never worked on user interfaces or with web services, so that this aspect of the project could be challenging for her. Additionally to that, she is not used to the software development Lifecycle, so that executing this project will include many new experiences to her.

5.2 Juan Garza

Juan Garza is a student at the RWTH University. He is currently in his 4th semester of studies towards a master's degree in computer science. During his studies, he deepened his knowledge of process mining by attending the lecture "Business Process intelligence". Moreover, he carried out a seminar on "Selected Topics in Process Mining". He is familiar with Java and Python as programming languages. Besides school projects, he has no programming experience in "real world" environments which may represent a difficulty for him.

5.3 Roles

6 Project office

6.1 Description

References

[1] van den Biggelaar F.J.H.M. van der Aalst W.M.P. Lu X., Fahland D. Handling duplicated tasks in process discovery by refining event labels. In *La Rosa M., Loos P., Pastor O. (eds) Business Process Management. BPM 2016. Lecture Notes in Computer Science, vol 9850. Springer, Cham*, 2016.