Process mining in programming game logs to differentiate between skill levels

Nick Richter
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
n.g.richter@student.utwente.nl

ABSTRACT

People have different skill levels in a certain expertise because some people have a talent for it or have worked hard to master a certain procedure. This paper will try to use process mining to see if skill level can be correctly assessed by letting a person play a game where a certain underlying knowledge and expertise is required. The game used in this paper is called Blockly Games by Google. This game teaches young children how to program and to think like a programmer using visual blocks. The way a person plays this game can say a lot about his or her skill in programming.

Keywords

Process mining, games, skill level, classification, user behaviour analysis

1. INTRODUCTION

Many people play games, they do it to relax or to experience the euphoria when they beat something 'impossible'. Some games, however, are made for you to test your skill in a certain profession. They have difficulty levels that get more difficult the more you advance. While such a game could be a good test for an employer looking for a future employee, this employee could 'brute-force' his way through the problems to prove his worth. Process mining could prevent this by looking at the whole process of the employee and giving him a grade based on how well he did during the problem solving, not only the end result.

Process mining is a discipline that sits in between data science and process science[1], which takes event logs as an input to discover, conform or even enhance process models. Process mining is different from data mining as it is both data-driven and process-centric[2]. This means that the data will follow a specific order of events all with a certain time stamp.

This paper will use process mining on event logs generated from a programming game to predict the level of skill a player has and to answer the question which parameters can be used to determine if a player is skilled or not. The way process mining will be used in this regard is to generate models and then analyze performances. The reason why process mining is being used is because computers are becoming much better at determining patterns in what seems like random data, and are therefore better in assessing the performance of an individual.

First this paper will put forth the research questions. Then the paper will give some background information into process mining and the game that will be used. After that some related work that this paper will build upon. Finally the method, results, conclusion and discussion will follow.

31th Twente Student Conference on IT July 5th, 2019, Enschede, The Netherlands.

2. RESEARCH QUESTIONS

The research questions for this paper are as follows.

RQ1 Can process mining distinguish between skilled players and unskilled players in a programming game?

- **RQ1.1** How is skill defined? What parameter(s) can be used to define skill?
- **RQ1.2** What algorithm(s) can be used to classify the results?

3. BACKGROUND

3.1 Process mining

Process mining is process-centric. It means that the events generated by the game have a certain order and happen at a specific point in time. These events should therefore have multiple necessary properties. Firstly, an event should always belong to a case and should be identified by the "case-id". The case is the overarching instance of what is connected to each other. Secondly, a task has to have happened. This means that an activity was performed at a certain time and should therefore contain an identifier for that activity and a time stamp[17]. Next to these mandatory properties an event can have additional information to make certain aspects more clear, like the person who performed the task or with which tool it was done.

3.2 Tools

There exist multiple tools to process event logs, ProM[1], Disco[2] and many more[3]. These tools are widely used to generate and improve process models or find bottlenecks in processes in businesses[4]. Another tool exist called RapidMiner. RapidMiner is a data science software platform with a lot of build-in functions to aid data analysts in predictive analysis, machine learning and much more. RapidMiner has a plugin called RapidProM[15], made by the

[1] http://promtools.org/
[2] https://fluxicon.com/disco/
[3] https://en.wikipedia.org/wiki/Process_mining#Software_for_process_mining
[4] https://www.hspi.it/wp-content/uploads/2017/11/HSPI_Process_Mining_Database_v1.1-Nov_17.pdf

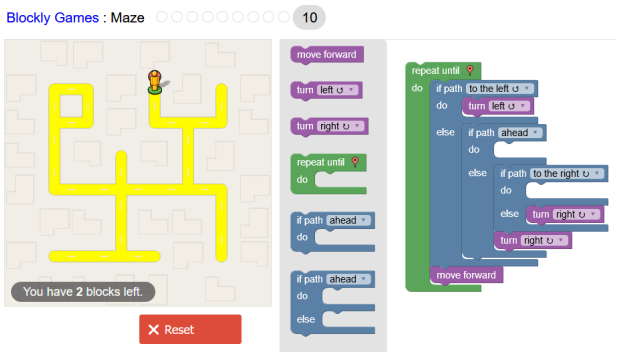

Figure 1. Level 10 Blockly Maze with 1 of many possible solutions

same people who made ProM, which makes it possible to use processes in RapidMiner.

3.3 Blockly Games

Blockly Games[5] by Google will be used as the programming game, as it is open source and can easily be modified. The game lets players drag blocks around that all have a certain programming relation as is shown in figure 1. The game does not, however, produce any event logs, so these will have to be coded in.

4. RELATED WORK

4.1 Process mining

Process mining has been used in several industries, from health care[11] to insurance[14] and logistics[4]. But process mining can be applied to software too[12]. Rubin et al. show that by having a server or client application log all events they could conclude many things about the usage of the software and where the errors or bottlenecks are. Liu[10] gives good examples on how to automatically generate models from software execution logs.

There is no related work that could be found on the particular topic concerning process mining in programming games. There has been, however, some research into data mining in game event logs[18][16].

A paper has been submitted to SIMPDA 2018 about process mining of events logs from Windows[6]. They showed that it can be done and used for multiple applications, mainly to monitor resource usage.

There has also been a recent study on the usage of process mining in helping evaluate the performance of an assessment[3]. While this research was performed with a case study in the educational sector it is of value for this paper as they show how they evaluate the performance of students based on event logs.

In the paper of Baykasoğlu et al.[3] they have a teacher use the software and have him perform the ideal combination of actions. They then compare the event log from the performance of the students with that of the teacher to find similarities. The more the event log differs from the ideal combination the lower the similarity score and the lower the overall grade the student gets. It is shown that the calculated grade based on this performance comes close to the manual grade the teacher gives the students as can be seen in figure 2.

How this paper uses process mining could be a good way to approach the question in this paper. One thing to note

[5]https://blockly-games.appspot.com/

	Key 1	Key 2	Key 3	Key 4	Key 5	Automatic marking (final grades)	Instructor's manual markings
Student 1	0.9692	0.9692	0.9692	0.9769	0.9376	97.6923	97.5
Student 2	0.6186	0.6186	0.6186	0.6186	0.5792	61.8552	55.0
Student 3	0.6253	0.6407	0.6330	0.6330	0.5937	64.0724	65.0
Student 4	0.6100	0.6253	0.6176	0.6176	0.5706	62.5339	70.0
Student 5	0.7136	0.7136	0.7136	0.7136	0.6665	71.3575	70.0
Student 6	0.6724	0.6801	0.6801	0.6724	0.6253	68.0090	70.0
Student 7	0.6416	0.6416	0.6416	0.6416	0.5946	64.1629	65.0
Student 8	0.7357	0.7357	0.7357	0.7357	0.6887	73.5747	75.0
Student 9	0.8059	0.8136	0.8136	0.8059	0.7511	81.3575	85.0
Student 10	0.6262	0.6262	0.6262	0.6262	0.5792	62.6244	70.0
Student 11	0.7991	0.7991	0.8068	0.7991	0.7520	80.6787	78.0
Student 12	0.4217	0.4217	0.4294	0.4294	0.4500	45.0000	48.0
Student 13	0.6665	0.6588	0.6588	0.6665	0.6271	66.6516	65.0
Student 14	0.5946	0.6100	0.6023	0.5946	0.5552	60.9955	60.0
Student 15	0.7204	0.7281	0.7204	0.7204	0.6810	72.8054	70.0

Figure 2. Automatic grades vs instructor's grades[3]

would be the randomness in certain games where there is no standard ideal combination of moves. Randomness could be eliminated or the computer could generate the perfect path the player should follow and compare it to the event log of the player.

4.2 Clustering

To know if a player belongs to a certain skill level the results have to be clustered. A couple related works could be identified on clustering. A recent master thesis on predicting user loyalty in a web application[7] has shown multiple ways how to cluster data albeit not accurate with the data he got.

Another study looked at if user behaviour could be clustered based on web log data using an improved K-means clustering algorithm[9].

Yet another study on user's skill level in an application using high frequency interface events (mouse movement, clicking, etc.) concluded that skill level can be correctly determined with a small error margin[8].

Most studies in this field focus on user interface proficiency and how to improve it. The aim of this study is to determine the skill level of the player in the underlying concepts of the game.

5. METHOD

5.1 Generating event logs

Blockly Games by Google is a free open source game that is based on the Blockly library, also by Google. The source code for this game is available at Github[6]. Because Google may not use the Windows operating system because of among other things security reasons, this project is only compilable on Linux-based operating systems. Windows WSL (Windows Subsystem for Linux) has been used to simulate a Linux environment (Ubuntu 18.04 LTS[7]) to successfully compile Blockly Games on a Windows machine.

Of all games that are contained in Blockly Games only the Maze game was selected for this research. This decision was made to limit the scope to a single game so research could be done more in depth. The reason the Maze game was selected was because it was the first game that introduced essential programming functions like the if-statement and while-loop.

Because Blockly Games does not generate log data on its own it had to be coded in. Blockly Games is made entirely in the JavaScript programming language but it still

[6]https://github.com/google/blockly-games
[7]https://www.microsoft.com/p/ubuntu-1804-lts/9n9tngvndl3q

has to be generated by a python script. For this generation nothing special had to be coded in Python and was automatically done, the actual code could be written in JavaScript. A function was identified in the main JavaScript file that was called whenever the internal work space had been changed. A change could be when a person selected a block in the work space or when a block was added, removed or dragged across the screen. An example of how the different actions were called was given in the code, which made it easier to find where modifications were needed.

A list was created in the JavaScript file to store the event log. Every time an event happened an object was appended to this list. The event objects were in the form: {Event: (...), Info: (...), Timestamp: (...), Level: (...)}. The event parameter specified what happened, all possible values were:

1. Starting Level
2. Modified Blocks
3. Executing Level
4. Resetting Level
5. Finished Level
6. Failed Level

The value of the info parameter depends on the event. In case of Modified Blocks it contained the work space representation. For the other events it did not contain usable information. The time stamp was generated by JavaScript with the new Date().getTime() method which gives back the milliseconds since Unix Epoch, the 1st of January 1970 00:00:00.

No distinction was made between adding, removing or dragging a block, or which block was being added, removed or dragged. The action of selecting of a block was not counted towards the event Modified Blocks as nothing was modified.

Prior to starting the game a questionnaire was shown to the players. These questions would give an indication of how skilled a player should be in the game. Questions included asking for their age, current study or highest study completed, number of lines of code written and any visual programming experience.

After every level the page refreshed and any data objects were reset. To fix this issue all data was saved to the browsers persistent storage, window.localStorage(). This object would not reset when the page refreshed or changed.

5.2 Gathering event logs

After modifying the code so event log data could be generated the next step started with the gathering of event logs. It became clear rather quickly that a website was needed with a URL so people could play the game on their laptop or computer. Individually requesting people to play the game on a specifically prepared laptop would become a time-sink and did not scale. Because Blockly Games was built by Google it uses the Google Cloud platform for hosting. This was already pretty much pre-configured and became the choice to host the modified game on.

The Google Cloud platform consists of many services and tools, including services for hosting, computations, data storage, data analysis and machine learning. Only hosting the modified game was not enough because the generated

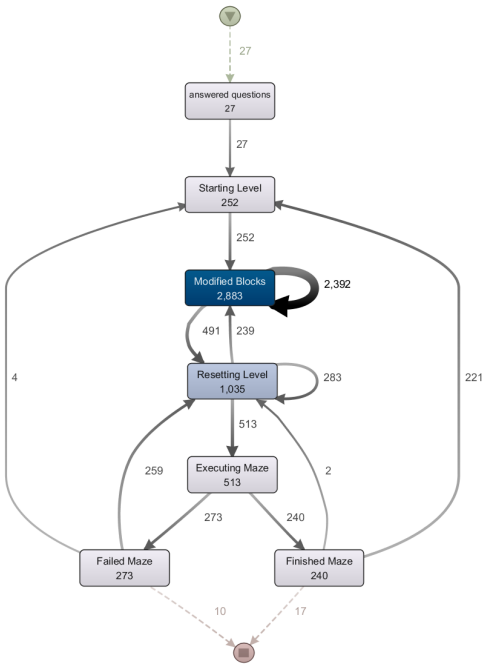

Figure 3. Process model of Blockly Games made with Disco

logs files should be stored somewhere. After researching the data storage services of Google Cloud, Google Datastore was chosen because of the example files already used by Blockly Games. Log files are send over from the client's browser to Google Datastore every time the user executed a level instead of every modification to the event logs to prevent the over saturation of the internet connection of the user.

5.3 Pre-processing

Google Datastore does not allow for easy extraction of the data. Luckily, when requesting the data in your browser (Google Chrome only as Firefox would return an error when you copy the whole file) a JSON file is being downloaded with all the information in it. Using your browsers network developer tool this data can be copied and saved in a text file. The data is still, however, in an unusable form for process mining tools. A python script was written to separate all the actions and place them in the right order for every user as is seen in Table 1. Time stamps are consistent for each user, only not between each player which is not a requirement.

5.4 Process Mining Tools

ProM and Disco were initially used for process mining but no suitable function and plugin could be found in helping to answer the research questions. Disco's user interface and generated models looked a lot nicer than ProM but had less overall functionalities. RapidMiner was instead used after no progress could be made in other tools.

In RapidMiner with RapidProM a petri net model was generated out of the process logs and given to a performance function together with more event data. This delivered a model with performance data included.

RapidMiner also includes 2 other functionalities, namely TurboPrep and AutoModel. These 2 functions were used

User	Maze	ActionID	Action	Timestamp	Info
1	0	0	Answered Questions	29-05-19 15:31:53	...
1	1	1	Starting Level	29-05-19 15:31:53	...
1	1	2	Modified Blocks	29-05-19 15:31:53	...
1	1	3	Resetting Level	29-05-19 19:54:38	...
1	1	4	Executing Maze	29-05-19 19:54:38	...
1	1	5	Failed Maze	29-05-19 19:54:38	...
1	1	6	Resetting Level	29-05-19 19:54:38	...
1	1	7	Modified Blocks	29-05-19 19:54:42	...
1	1	8	Resetting Level	29-05-19 19:54:44	...
1	1	9	Resetting Level	29-05-19 19:54:45	...
1	1	10	Executing Maze	29-05-19 19:54:45	...
1	1	11	Finished Maze	29-05-19 19:54:45	...
1	2	12	Starting Level	29-05-19 19:55:00	...
...	...	...	...	...	...
20	10	3516	Executing Maze	29-05-19 12:58:45	...
20	10	3517	Failed Maze	29-05-19 12:58:45	...
20	10	3518	Resetting Level	29-05-19 12:58:45	...
20	10	3519	Modified Blocks	29-05-19 12:59:23	...
20	10	3520	Modified Blocks	29-05-19 12:59:24	...
20	10	3521	Resetting Level	29-05-19 12:59:28	...
20	10	3522	Resetting Level	29-05-19 12:59:29	...
20	10	3523	Executing Maze	29-05-19 12:59:29	...
...	...	...	...	...	...

Table 1. Generated event log from Blockly

to do some general data mining on the event logs. Data was aggregated in TurboPrep into a usable format for data mining and then given to AutoModel. AutoModel is a function made for clustering, prediction or detecting outliers. For the prediction function of AutoModel at least 100 rows of data is needed to train the algorithm accuratly, according to RapidMiner.

6. RESULTS

6.1 Gathered Data

The link to the website was sent to a diverse group of people, consisting of Technical Computer Science students, family and friends of the researcher. It was decided to not publish the link to a public internet place for fear of attracting spam replies, invalidating the research. Some data was incomplete which indicated that some people closed the browser session and did not fully complete the game. In total 31 responses came in, most of which were from Technical Computer Science students. 4 of these responses were empty responses and were discarded during the pre-processing stage. Responses that did start but not finish the game were included as some players were not proficient enough to complete all the levels. For example, one response tried levels 5 to 8 but could not complete them and did not start levels 9 and 10.

6.2 Parameters

The definition of skill that will be used for these results is as defined by Merriam-Webster[13]:

Definition 1. *The ability to use one's knowledge effectively and readily in execution or performance.*

In the Maze level of Blockly Games the skill that is being tested is effectively solving the maze using programming knowledge.

A number of parameters could be thought of that would indicate sufficient skill in the game. One of them is if a person could finish the levels. If someone is not able to finish the levels, especially looking at level 10, it indicates they did not have the needed insights and knowledge to successfully complete it and are therefore less skilled.

Another parameter for determining skill is the number of times an action was performed. If a person has to move a lot of blocks this might indicate that this person does not know what the fastest way to complete the level is. It might also be that this person does not know how to navigate around in the game, this indicates another proficiency that the player lacks and should be considered if it is part of the skill being tested or not.

One parameter that might be closely related to the one previously mentioned is the amount of time taken to complete the levels. If the user performs more actions it naturally takes more time but this parameter is also important if the amount of blocks moved is low. This indicates that this person is taking their time thinking and means that he or she does not readily apply his or her knowledge. A person can also be really fast or really slow in dragging the blocks, so this parameter should be used along side other parameters to classify skill.

A fourth parameter to determine if a person does not have enough skill is the amount of failed executions, especially for the last maze level. If a person has many failed executions it might indicate that this person is brute forcing its way through the levels. He or she could do so very fast but with many failures which is why this parameter should be included.

All these parameters can be measured by the computer and subsequently put into an algorithm. One parameter that is not of this kind is the thinking pattern of the player. For this parameter an expert is needed to grade the player on how well he or she does. Prior to started the games players were asked questions indicating their programming experience. This is not enough when wanting to know how they do things, as self-assessment is generally not accurate.

A parameter that cannot be used in Blockly Games that might be used in other games is the path a player took and the end result that player produces to complete the level. In Blockly Games and possibly other programming related

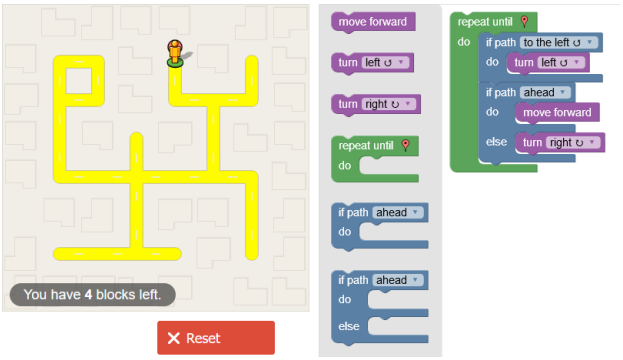

Figure 4. Level 10 Blockly Maze alternative solution

games there is no one ideal path. Depending on the way some people have been taught to program they could handle a problem in a different way. A player could start by building the blocks from the beginning or by building all the sub-groups of blocks first before dragging them inside a while loop.

The end result the player produces may also be different and may not be faster in execution but may be nicer to look at or faster to implement. These different ways are all good if the level completed successfully. Figure 4 shows a different solution than figure 1 even though the end result is the same.

Herein lies a problem, if the process of what players do in Blockly Games can not be used as a parameter, then process mining may not be used to assess skill levels. Previously mentioned parameters that can be used in assessing skill levels can all be assessed with regular data mining techniques such as clustering or machine learning models.

6.3 Algorithms

As was mentioned before, RapidMiner comes with a build-in function AutoModel. For the clustering sub-function two methods are implemented, the k-Means Clustering algorithm and the x-Means Clustering algorithm. ProM has a couple other clustering algorithms but no way to turn a process log into a usable data set for a clustering algorithm. These features fit more into regular data mining approaches than process mining as it uses aggregated data from the process log and removes any process associated information.

Figure 7 shows the aggregated data and classifies them into 3 clusters. Not much can be said about this graph as it has not taken into account the amount of levels played, only the amount of executions and failed executions. If you do not play that many levels then the executions will also be lower. Figure 6 shows the correlation between the amount of modified blocks and failed execution attempts. You can clearly see that they roughly have the same dips and peaks. These graphs are only for showing what it can look like using clustering approaches, no result is being drawn from them.

There are a couple more prediction algorithms in AutoModel. These prediction algorithms can be trained on real data that is already classified, when a non-classified data entry is put into the algorithm it can classify it. Some possible algorithms are: Naive Bayes, Generalized Linear Model and Deep Learning. As was already mentioned, a minimum of 100 rows of data are required. This is one of the two reasons why this was not being researched, with the other being bad manual classification. This approach of predicting skill levels shows promise but was not further researched and could be a future research topic.

If process mining could be used then similarity algorithms such as the similarity analysis shown in table 1 of Baykasoğlu et al.[3] can be used to indicate how similar the processes are and can then be graded. Processes similar to an expert or just the best possible path are then scored really high and processes with many detours are scored really low.

6.4 Process Mining in Programming Games

The answer to research question 1, if process mining can distinguish between skilled players and unskilled players in programming games, is answered with no. Blockly Games was chosen as a case because of it being open source. But it allows for many different process flows for which not one can be chosen as the best one. This in turn does not allow process mining to distinguish between skilled and unskilled players via this way.

Generating a petri net in RapidMiner with the RapidProM plugin and subsequently analyzing the performance of event logs on it did not result in a good graph, this is possibly a fault in RapidProM or in the data. Figure 5 gives the output of this analysis but no useful information can be extracted from it as the output seems to be randomly generated and does not coincide with anything in the data set.

What can be said based on this research, is that some other data mining approaches show potential in classifying players as skilled or unskilled by first aggregating the data. Players can be clustered based on how many times they executed a level or how many times they modified a block. Manual classification is then needed to specify which clusters are the beginners and which are the experts. A prediction model can also be build for this data. Better manual classification is needed for at least 100 data subjects. This is the minimum lower limit RapidMiner has placed on this feature.

7. DISCUSSION

As was shown in the results section the method which is used in this research could not distinguish between skilled and unskilled players in Blockly Games. Oversights in this paper and possible future research will be discussed in this section.

The event log was possibly too homogeneous, the only actions according to the process log a player could do was ”starting a level”, ”modifying a block”, ”executing the algorithm” and ”resetting the level” if the execution failed. These were not all the specific actions the player could do as modifying a block could be adding, moving, removing or changing a block. There were a total of 10 different blocks in this game of which most have the 4 actions just specified. This would cause the process log to be much more rich in data and possibly allow process mining algorithms to find certain patterns indicating skillfulness.

Another possibility is that this problem is better suited for other data mining techniques, not process mining. If you want to distinguish between skill levels you want to classify players, this is a well researched subject in data mining[5][19]. RapidMiner is a data mining tool and allows aggregating huge process logs into smaller data sets to classify certain parameters. Process mining is better suited for the discovery of a process model and subsequently figuring out how to make it better.

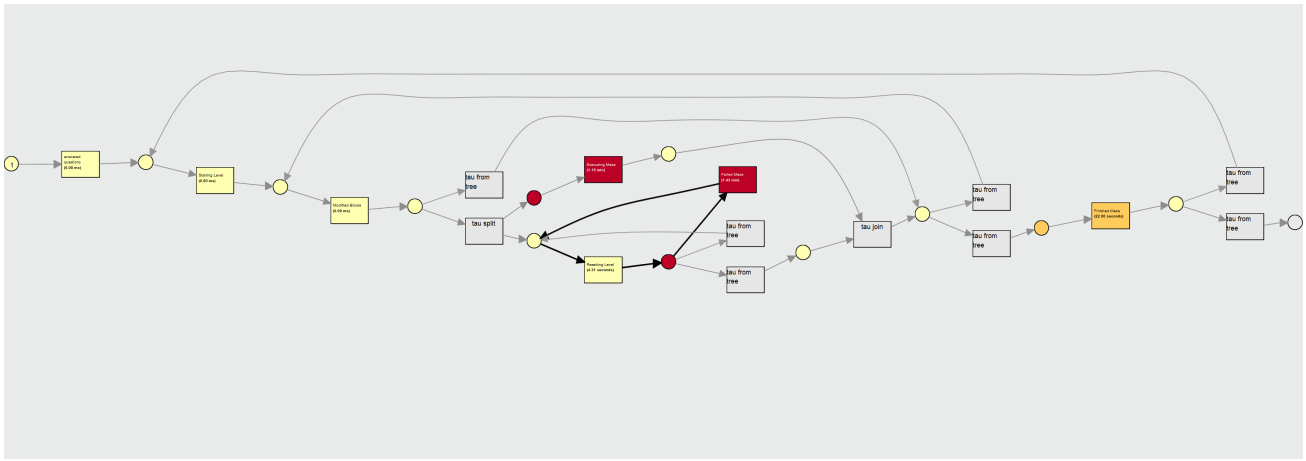

Figure 5. Output of RapidProM 'Analyze Performance' function

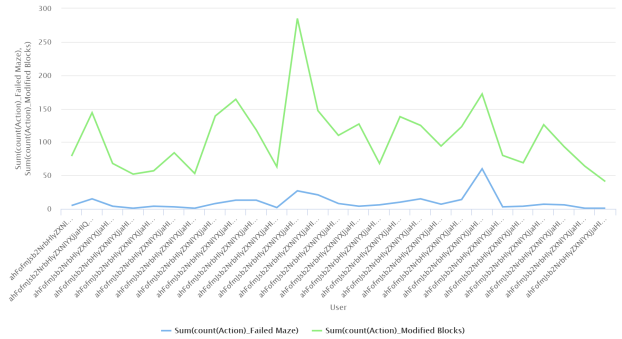

Figure 6. Correlation between number of modifications and failed executions

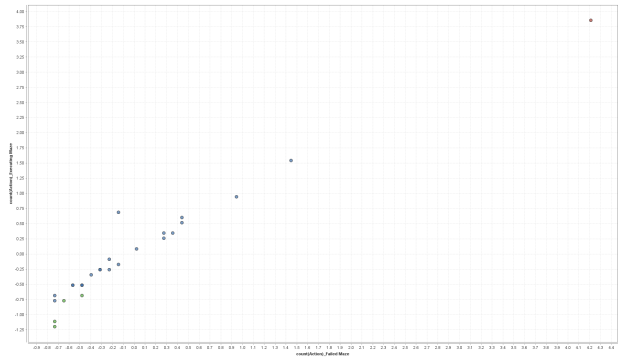

Figure 7. k-Means clustering with 3 clusters

In a game such as Blockly Games and with programming in general there is no one good way to do things. There is no best path that could be followed, every player thinks a bit differently and might come to different solutions that all are valid. This makes process mining in programming related games much harder. A follow up research could look at games that have a more fixed path to a solution and that every detour from that path is considered inefficient.

The prediction feature of the AutoModel part of RapidMiner wants at least 100 rows of data to train and test the algorithms. In this research only 31 responses came in of which 27 were sufficient. Follow up research would want to have at least 100 rows of data to use this feature and to confidently say that the algorithm has a certain accuracy. Another problem is that those 100 rows of data would need to be already classified with a skill level. A simple questionnaire would not work as people grossly under- or overestimate their programming knowledge. A tutor will need to watch over the shoulders of the person and see how they are performing the tasks and give them a grade.

8. CONCLUSIONS

In this paper, it has not been shown that process mining can distinguish between skill levels in programming games such as Blockly Games because of the wide range of paths to successfully complete the levels. Some possible solutions have been suggested for further research, namely looking at other data mining approaches and finding another sort of game with a more fixed ideal path solution. If more research into process mining in programming games is wanted, possible solutions would be gathering richer data and more data (at least 100 data subjects). Also better manual classification by a tutor is needed for machine learning to be trained to be more accurate.

It has been shown how process logs have been generated in Blockly Games and automatically gathered, this could be used for other purposes related to process mining. Possible parameters in Blockly Games on which to judge skill level have also been brought forward.

9. REFERENCES

[1] W. v. d. Aalst. *Process mining: discovery, conformance and enhancement of business processes*. Springer, Berlin, 2011. OCLC: 846443171.

[2] W. v. d. Aalst. *Process Mining: Data Science in Action*. Springer-Verlag, Berlin Heidelberg, 2 edition, 2016.

[3] A. Baykasoğlu, B. K. Özbel, N. Dudaklı, K. Subulan, and M. E. Şenol. Process mining based approach to performance evaluation in computer-aided examinations. *Computer Applications in Engineering Education*, 26(5):1841–1861, 2018.

[4] T. Becker and W. Intoyoad. Context Aware Process Mining in Logistics. *Procedia CIRP*, 63:557–562, Jan. 2017.

[5] P. Berkhin. Survey of Clustering Data Mining Techniques. page 56.

[6] R. Dolak, M. Janakova, and J. Botlik. Process mining of events log from Windows. In *Proceedings of the 8th International Symposium on Data-driven Process Discovery and Analysis (SIMPDA 2018)*, Dec. 2018.

[7] K. M. El Assal. Predicting user loyalty in an education support web application based on usage data. Master's thesis, University of Twente, Apr. 2018.

[8] A. Ghazarian and S. M. Noorhosseini. Automatic detection of users' skill levels using high-frequency user interface events. *User Modeling and User-Adapted Interaction*, 20(2):109–146, June 2010.

[9] S. P. Karthik and A. Sheshasaayee. Clustering of user behaviour based on web log data using improved K-means clustering algorithm. *International Journal of Engineering and Technology*, 8(1):305–310, Feb. 2016.

[10] C. Liu. Automatic Discovery of Behavioral Models From Software Execution Data. *IEEE Transactions on Automation Science and Engineering*, 15(4):1897–1908, Oct. 2018.

[11] E. Rojas, J. Munoz-Gama, M. Sepúlveda, and D. Capurro. Process mining in healthcare: A literature review. *Journal of Biomedical Informatics*, 61:224–236, June 2016.

[12] V. Rubin, A. Mitsyuk, I. Lomazova, and W. M. P. Aalst. Process Mining Can Be Applied to Software Too! In *The 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, At Torino, Italy*, Sept. 2014.

[13] Skill. *Retrieved at https://www.merriam-webster.com/dictionary/skill*. Merriam-Webster, on 04-06-2019.

[14] S. Suriadi, M. Wynn, C. Ouyang, A. Ter, and N. J. van Dijk. Understanding Process Behaviours in a Large Insurance Company in Australia: A Case Study. In *The role of conceptual modeling in enabling open innovation*, pages 449–464. June 2013.

[15] W. M. P. van der Aalst, A. Bolt, and S. J. van Zelst. RapidProM: Mine Your Processes and Not Just Your Data. *arXiv:1703.03740 [cs]*, Mar. 2017. arXiv: 1703.03740.

[16] B. G. Weber and M. Mateas. A data mining approach to strategy prediction. In *2009 IEEE Symposium on Computational Intelligence and Games*, pages 140–147, Sept. 2009.

[17] A. Weijters and W. M. P. Aalst. Workflow Mining: Discovering Workflow Models from Event-Based Data. In *Proceedings of the ECAI Workshop on Knowledge Discovery and Spatial Data*, pages 78–84, Jan. 2002.

[18] S. Wender. Data Mining and Machine Learning with Computer Game Logs. Project Report 4685895, University of Auckland, Oct. 2017.

[19] G. J. Williams, S. J. Simoff, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, and G. Weikum, editors. *Data Mining: Theory, Methodology, Techniques, and Applications*, volume 3755 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.