Game Achievement Analysis: Process Mining Approach

Martin Macak(✉), Lukas Daubner, Julia Jamnicka, and Barbora Buhnova

Faculty of Informatics, Masaryk University, Brno, Czech Republic
{macak,daubner,jamnicka,buhnova}@mail.muni.cz

Abstract. Data-oriented techniques are currently standardly used in the video game domain, providing an interesting insight into the players' behaviour. However, the game can be seen as a set of steps that are performed for its completion. Therefore, these steps form a process. Process mining is a discipline with a focus on process analysis which can help to bring additional insights to the analysts. Hence this work explores the potential of a process-oriented approach in this context. We chose the game achievement log as the dataset as it contains valuable information about the player's steps in the game. Furthermore, it is publicly available, and therefore, anyone, not only game developers, can perform the process analysis. The dataset and the analysis source code used in this work were made publicly available.

Keywords: Process mining · Game achievements · Data analysis

1 Introduction

As the popularity of video games rises [15], so does the amount of data showing player in-game behaviour [5]. The field of game analytics focuses on analyzing said data to improve game development and design. Data mining is commonly used for game analysis; however, in contrast to traditional data-centric approaches, process mining [1] involves both data and end-to-end processes in the analysis, offering unique benefits for the analysis results [6]. As the game can be viewed as the set of steps that are performed for its completion, process mining can be an ideal candidate to provide additional insights to the analysts about the players' progression through a game. We chose the game achievement log as the dataset as it contains valuable information about the player's steps in the game. The other benefit is that this log is publicly available, and therefore, anyone, not only game developers, can perform the process analysis.

This paper explores the potential of process mining analysis in this context. Therefore it demonstrates and discusses four identified use cases in game analytics, taking a process-oriented approach and utilizing the process mining techniques. We used publicly available achievement data from the Steam platform[1].

[1] https://store.steampowered.com/.

B. Li et al. (Eds.): ADMA 2021, LNAI 13088, pp. 68–82, 2022.
https://doi.org/10.1007/978-3-030-95408-6_6

Our focus is on the typical playthrough of players, comparing player behaviour from different groups, e.g., those who finished playing the game versus those who did not or players who rated the game positively versus negatively. The work also aims to discover bottlenecks, i.e., game parts the players spent the longest time advancing through, and suggest improvement of pace and difficulty based on the result. Detection of anomalous behaviour is also a subject of this analysis.

The paper is organized as follows. Section 2 provides basic understanding of the topic. Section 3 outlines the related work. Section 4 presents the data set, classification and choice of games for analysis. Section 5 demonstrates the considered use cases. It is followed by Sect. 6 which discusses the results and suitability of various types of games for the mentioned analyses. Finally, Sect. 7 concludes the paper.

2 Background

This section provides the technical background on process mining and achievements, to build the elementary understanding needed for the rest of the paper.

2.1 Process Mining

Process mining techniques [1] aim to understand and analyze processes, providing more data interpretation perspectives. They are helping with challenging tasks in many domains, such as healthcare [11], manufacturing [13] and education [10]. In this work, they are used to extract knowledge from achievement logs to provide a better understanding of the game playing process. Moreover, all its technique types are used in this work, namely 1) *process discovery*, which aims to find a descriptive model of the underlying process from event logs, 2) *conformance checking*, which inspects whether the execution of the process conforms to the corresponding reference process model, and 3) *process enhancement*, which, in our case, extends the process model with time perspective, allowing us to inspect bottlenecks.

2.2 Achievements

Game achievements (also called trophies) form a system of rewards in video games. They operate on a layer separate from the actual game [4] and have no effect on the in-game progress of the player [12].

The purpose of the achievements is to provide extrinsic motivation to ensure greater player participation [3]. Whether they have this effect and what possible problems their use can cause have been a subject of various studies [3,4,7].

While different platforms may present achievements in a slightly different form, some components are always present. Concretely, *Title*, a unique identifier of the achievement and an *Icon*, a visual representation of the achievement. Other common features are the *Description* which usually offers hints on how the achievement can be unlocked, and *Timestamp* specifying the date and time when the player unlocked the achievement. Additionally, *Points* can be awarded to the player after the achievement is unlocked.

3 Related Work

Data mining is suited for analysis of large datasets and allows for effective analysis of game telemetry data [5]. Several studies that focus on video games, and specifically on video game achievements, use data mining techniques for analysis.

Apperly and Gandolfi [2] use Steam achievements to analyze behaviour of players in the game NieR: Automata (PlatinumGames, 2017). Dividing the game's achievements into several categories, the authors discover the percentage of players that gained the individual achievements. Their focus is therefore purely data-centric and is not interested in the game as a process.

Heeter et al. [8] use achievements (among other variables) as game metrics to measure meaningful player behaviour; the goal is to analyze differences in gameplay of two types of players, specifically the fixed-mindset and growth-mindset players. While fixed-mindset players believe that gaming skills cannot be improved and avoid hard challenges, growth-mindset players believe that their skills can be developed further precisely by challenging themselves. The study confirms that there are significant differences in the in-game behaviour of the two types of players. However, the focus of the study is exclusively on learning games, as well as purely data-centric.

Siquera et al. [18] aim to deduce World of Warcraft (Blizzard Entertainment, 2004) player profiles by using various data mining methods. The results divide players into categories according to their game behaviour, based on their time spent playing the game. Furthermore, based on the relevant game metrics, the authors are able to predict the probability of a player renewing their subscription. This study does not use achievements as game metrics.

While data mining offers many benefits in the field of game analytics, it does not consider a *process*, i.e. a sequence of activities executed in order, in its analyses, nor is the focus of data mining on improving said processes [14]. Since player behaviour can be considered to form a process, process mining can be suited to various types of game analysis.

There are few studies focused on games that use process mining. Uhlmann et al. [19] focus specifically on non-digital learning games and board games, and therefore cannot consider automatically logged achievements in the study. Ramadan et al. [16] focus on logged gaming data; however, the data was collected from a game created specifically for the study, and it did not contain achievements.

4 Data Preparation

Achievement data of a large number of players is needed for any meaningful analysis of video game achievements. One of the ways to obtain the data is through Steam, a video game platform that provides APIs for downloading data. This section provides overview of the data set, its preparation and preprocessing. We made the used dataset, including its process mining analysis, publicly available[2].

[2] https://github.com/lasaris/Game-Achievements-for-Process-Mining.

4.1 Steam Achievements Extraction

Steam Web API offers interfaces and methods to download data from the platform. From the Steam APIs interfaces available, we use the `ISteamUserStats`[3], containing methods for querying game statistics data, including achievements.

However, Steam Web API does not contain any method that would return a list of users that have played a particular game. A solution we found is to make a Store API[4] call to download a list of user reviews of a game, since every review contains the Steam ID of the author. As the number of reviews is large, the results are paginated and accessed by the parameter `cursor`. The method also returns other statistics, e.g., the total time the user spent playing the game, the last time the user played the game, whether the review was positive or negative, and the text of the review itself.

4.2 Conversion to Event Log

In order for the data to be usable in process mining analysis, it has to have mandatory fields for an event log. Concretely, those are *case id*, *activity* and a *timestamp*, which need to be extracted from the achievement data.

The case id identifies the case, i.e., an instance of the process, and in our data corresponds to a user's Steam ID. The individual events that belong to a particular case are identified by the activity, represented by an unlocked achievement's name. Any commas in an achievement's name are deleted since we use a comma as a separator later. Finally, since the events need to be ordered, the timestamp is included, representing the time when achievement was unlocked.

The event data are saved to a CSV file in the following format:

```
steam_id,achievement_name,unlock_timestamp
```

Additional player data are saved into a JSON file in the following format:

```
<steam id>: {
    "playtime":              <num minutes played>,
    "left_positive_review": <left positive review>,
    "review":                <review text>,
    "collected_all":         <has all achievements>
}
```

4.3 Game Categorization

For the purposes of this study, we established three types of game categories that bear relevance to our analysis. The description of the categories are as follows:

Game Linearity. Games can be divided according to the linearity. We discern *linear games* which a clearly defined storyline that a player has to follow in a

[3] https://partner.steamgames.com/doc/webapi/ISteamUserStats.
[4] Details can be found at: https://partner.steamgames.com/doc/store/reviews.

given order, with little variation allowed. Another type is *slightly linear games*, which usually have a main storyline, but also allow greater freedom for players to explore (e.g., open-world games, or games where the players may follow divergent storylines). The last type, *non-linear games*, encompass all other types of games, e.g., games with no storyline or games that are played repeatedly in rounds.

This category is important to consider when attempting to discover its process model. A linear game should have a clear, concise process model with far less variation (since players must follow a clear-cut path). On the other hand, a non-linear game might produce a complex process model, which would be hard to understand, and therefore, more advanced processing of data might be needed.

Achievement Types. We divide achievements into *progress achievements* and *optional achievements*. Progress achievements are awarded for crossing a certain point in a game, usually finishing or starting a level; everyone that plays the game will be awarded these achievements so that they act as milestones of player progression. On the other hand, optional achievements are usually awarded for the completion of special tasks or side-quests, and only a certain percentage of players will be able to unlock these achievements. Therefore, the games can be divided into those that contain only progress achievements, those with only optional achievements, and that contain a combination of both.

This category is as important as the previous one. Even if a game is linear, the absence of progress achievements will make it more difficult to discover a fitting process model. Games with a combination of progress and optional achievements seem to be the most viable for analysis since the progress achievements will clearly mark levels/milestones, while the optional achievements bring enough variation to offer interesting information. The games with only progress achievements should have a simple and easily discoverable model. Although, variations are expected for the players that did not finish the game.

Rating. The rating of the game is determined by the amount of positive and negative reviews a game has received on Steam ranging from *Overwhelmingly Positive* to *Overwhelmingly Negative*. The rating is used as an indicator of popularity, which indirectly influences various factors, such as the percentage of people who finished the game or the number of reviews. Therefore it is expected that games with more positive rating should have more relevant and complete data for purposes of this study.

4.4 Selected Games

The games used in our analyses were chosen to represent different game categories. Games with a negative rating were not considered due to generally insufficient data. Within the paper, the games are referred to by labels G1–G8 specified in Table 1, which also contain their respective categories.

Table 1. Summary of selected games for the study, with their respective categorization.

	Game	Game Linearity	Achievement Types	Rating
G1	Gris (Nomada Studio, 2018)	L	P(5), O(12)	OP
G2	Hades (Supergiant Games, 2018)	SL	P(5), O(44)	OP
G3	TIS-100 (Zachtronics, 2015)	NL	O(10)	OP
G4	Per Aspera (Tlön Industries, 2020)	SL	P(5), O(27)	M
G5	Oxygen Not Included (Klei Entertainment, 2017)	NL	O(35)	OP
G6	Friday the 13th: The Game (IllFonic, 2017)	NL	O(53)	MP
G7	Witcher 3: The Wild Hunt (CD Projekt, 2015)	SL	P(8), O(70)	OP
G8	Black Mirror (King Art Games, 2017)	L	P(16), O(5)	M

L — linear, SL — slightly linear, NL — non-linear
P — progress achievements (count), O — optional achievements (count)
OP — overwhelmingly positive, MP — mostly positive, M — mixed

4.5 Data Filtering

Prior to the analysis, the event log is checked for errors caused by faulty event logging and filtered accordingly. Incorrect logging of the unlocked achievements can be caused by the Steam client, as by playing a game with Steam in offline mode can record wrong unlocking time or not logging at all. Another possibility is a bug in the game or dishonest player using a tool[5] to unlock Steam achievements without unlocking them in the game.

Within this study, three rules are considered. Cases that matches any of the rules are filtered out. However, it might be desirable to analyze these traces further. While games without progress achievements cannot be easily checked for these types of errors, they could also contain erroneous cases. A solution to this problem is suggested in Sect. 5.4. The rules are:

- Progress achievements that were unlocked out of order. Since the progress achievements mark passing a level in the game, they should be unlocked in the same specific order by every player.
- Progress achievements that were unlocked at the same time. While game can allow for multiple achievements to be unlocked at the same time, this should not be the case with progress achievements.
- Achievements with timestamp to 2008. Since achievements were introduced to Steam in 2008 [9].

5 Analysis of Game Achievements

In this section, the four use cases of process-focused game analytics utilizing process mining are demonstrated. These are of particular interest for game designers offering valuable insight into player behaviour, and should serve as an inspiration

[5] For example: https://github.com/gibbed/SteamAchievementManager.

for analyses in the game industry. Each analysis is performed in Python using the PM4Py[6] library.

5.1 Typical Playthrough

Typical playthrough refers to the mainstream behaviour of players in a game. *Playthrough* can be defined as the act of playing a game from start to its end. Since some games do not have a clearly defined ending point, the playthrough in this context is the act of playing a game from the first achievement unlocked to the last achievement unlocked.

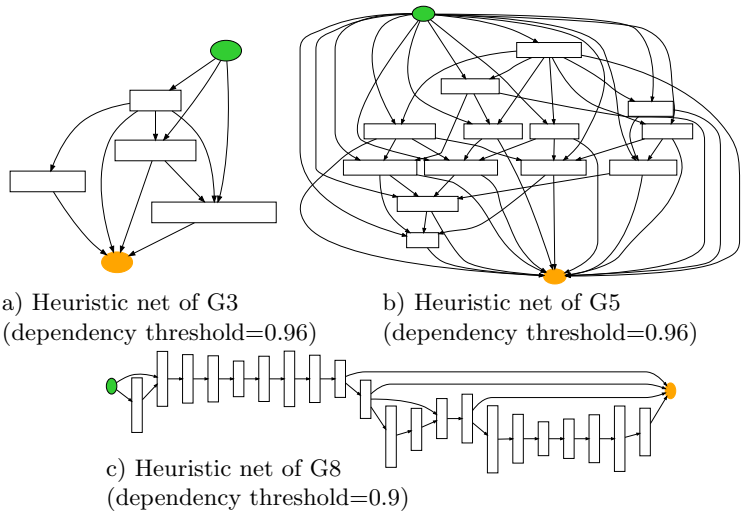

Fig. 1. Process models discovered by Heuristics miner.

A selection of the discovered process models is shown in Fig. 1 (their nodes and transitions do not contain captions because of readability issues). Noticeably, the discovered process models of games G3 and G5 look vastly different from the process model of G8. As G8 is a highly linear game, the process model is straightforward and easy to understand. Interestingly, the behaviour of players as shown in the G3 model is less straightforward despite G3 being a linear game. This is caused by the absence of progress achievements, as G3 contains optional achievements exclusively. Furthermore, it is interesting to point out that even though G5 is a non-linear game, its process model does not contain any cycle, which means the achievements have some virtual ordering.

For the discovery of typical playthrough Heuristic miner was used, with dependency threshold adjusted accordingly so that the infrequent behaviour is

[6] https://pm4py.fit.fraunhofer.de/.

filtered out. The threshold also has to be adapted according to the linearity of a game. However, it seems setting the value at 0.96 produces a fitting process model for the majority of games. Concerning further parameters, the minimum activity count was set to 33.3% and the cleaning threshold to 0.5.

Process discovery can also help us with getting the most frequent trace variant in an event log. It can be discovered by using the variant filter and setting the percentage parameter to 0 (only the most frequent variant is kept). In the case of G1, the most frequent variant corresponds to a full playthrough of a game, which is an exception when compared to the other games. It shows that a high percentage of players (∼17%) had exactly the same playthrough. On the other hand, the most common variant of G2 contains only the first progress achievement. However, since it represents only 18 players (<2%), it cannot be considered as representative behaviour. Instead, it means that the optional achievements were unlocked in a more random order that cannot be simply generalized. However, while the most common variant of G3 also contains only the first progress achievement, it actually represents the common behaviour well, since 204 of players (∼30%) truly obtained only a single, trivial achievement.

5.2 Comparing Player Behaviour

Player behaviour can be compared in multiple important aspects, e.g., whether they finished the game or not, whether they have a positive or negative experience from the game, and based on the version of the game they played. In this demonstration, we show the first case.

We compare the behaviour of players who finished the game with the behaviour of players that did not because it is useful to have insight into the reasons why players quit the game before the finish. Two different event logs are prepared by including/excluding traces that do not contain the game's end achievement, and two process models are discovered on them. Additionally, the event log containing the game's end achievement was further filtered to contain only achievements gained before reaching the end of the game to exclude behaviour on replays.

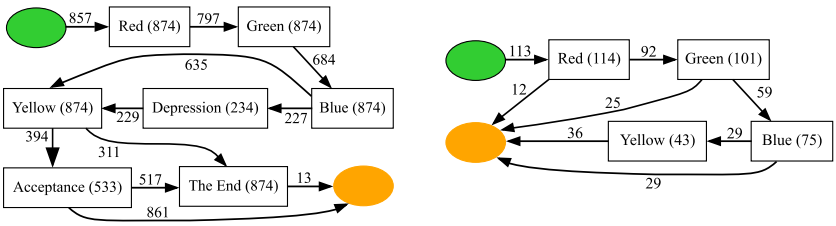

Fig. 2. Comparison of two process models of game G1, the left one showing behaviour of players that finished the game, while the model on the right shows behaviour of players that did not. Dependency threshold was set to 0.96 and cleaning threshold was set to 20% for both models. (Color figure online)

Two such process models of G1 are in Fig. 2. There are significant differences between the finished model (model of finished traces) and the unfinished model (model of unfinished traces). The first observation is before which achievements the players quit in the unfinished model. As players get further in the game, the probability that they will not finish the game rises, which is depicted by the transitions from the individual level achievements to the end node of the model. While only 12 of the 114 players quit after achieving *Red*, i.e., during the first level, the number doubles in the second level and triples by the last level. Interestingly, unlocking optional achievements was not common among the players who quit, suggesting they did not explore the game as much as players who finished playing it.

Another observation are the interplay of optional achievements. Concretely, *Depression* was commonly unlocked after *Blue*, i.e., during the third level, while *Acceptance* was unlocked as the last achievement before the end achievement, i.e., during the ultimate level. Since we discovered that 75 quitting players reached *Blue* and only 43 quitting players reached *Yellow*, it can be assumed that the low percentage of unlocked optional achievements was caused by the loss of players in the former levels.

Note that even though we filtered the events to contain only traces with the *The End* achievement and filtered out the achievements unlocked later, the process model of the finished game ends in 13 cases with *Acceptance*. It is caused by the threshold parameters of the discovery algorithm that were used, specifically in cases where there are many activities between *Acceptance* and *The End*.

5.3 Game Level Analysis

Good level design is important part of a game's success [17]. This section of the analysis focuses on discovering bottlenecks in the game, i.e., finding out which game levels took players the longest time to pass. Additionally, a correlation between level duration and the number of players that quit the game during the respective levels is explored. Since this analysis focuses on game levels, only games with progress achievements (i.e., achievements that are unlocked by passing through a certain point in the game) are suitable.

Finding Bottlenecks. For the bottleneck analysis, the models need to be enhanced with information about *performance* by an aggregate function of the elapsed time between the unlocking of two achievements. Two functions are applicable, the mean and the median. However, using the mean lengthens the duration significantly since the few players who took month-long breaks from the game skew the numbers. The median produces the more suitable level duration that corresponds closely to the actual time players spent passing the level.

Since only the level duration is relevant, the game event log can be filtered by keeping only the progress achievements. The time between two achievements in the model corresponds to the time spent in the corresponding level. For the filtered event log, a directly-follows-graph is sufficient and supported by PM4Py for performance visualization.

The chosen models are in Fig. 3, where the transitions between activities, i.e., levels' starting points, contain information about the median level duration. The activity duration does not apply since achievements contain only the time when they were unlocked. Note that the first progress achievement actually marks the start of a second level, not the first one, since none of the games contains an achievement signifying the game start. In this case, determining the duration of the first level is not possible, but for the sake of clarity, the actual second level is referred to as the first one when describing models.

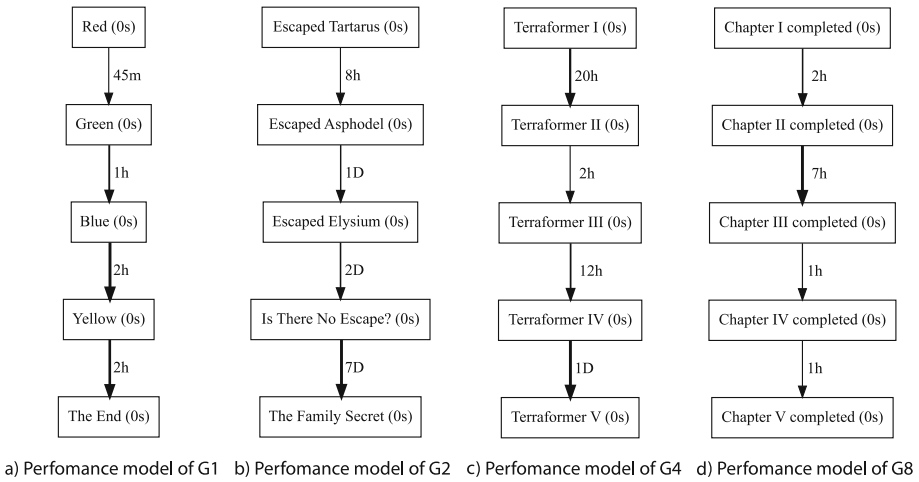

Fig. 3. Performance models showing the median level duration.

By analyzing the models, G1 shows the least amount of variation in the level duration. The first level *Red* lasting 45 min and the last level *Yellow* lasting 2 h[7], with the last two levels being very slight bottlenecks in comparison. A clear upward trend in the level duration can be observed. The same upward trend can be seen in G2. However, the variation in the level duration is more pronounced, ranging from 8 h in the first level to the 7 days in the last level (longest duration in all games). The clear bottleneck in the game is, therefore, the level starting with the achievement *Is There No Escape?*, and to a much lesser degree, the level preceding it, with the median duration of two days.

Neither G8 nor G4 show the upward trend of the level duration. G8 contains a bottleneck in the level preceding the *Chapter III completed* achievement, with the level duration at 7 h, a major increase when compared to the other levels. The last game, G4, does not conform to the trend of the first levels' relatively short duration that can be observed in the other games. While slightly shorter than the last level, the first level can also be considered a bottleneck with a median

[7] At the time of writing this paper, PM4Py cannot adjust the duration accuracy.

duration of 20 h. The last level starting with the *Terraformer IV* achievement is the largest bottleneck with a 1 day duration.

Correlation with Player Statistics. If the bottlenecks correspond to the levels where most players left the game, it can signal a bad level design. Cross-checking the findings with negative reviews offers even more information about the individual game levels.

Particularly interesting is G4, which breaks the pattern of all of the other games, specifically the increase of players quitting the game at every further level. Instead, 30% and 36% of players stopped playing in the first and the last level, respectively, which is markedly more than the other two levels. This corresponds exactly to the two bottlenecks found in the process model in the first and the last level, suggesting that the players experience difficulties in these two levels.

It is further corroborated by the number of negative reviews. More than half of the players who stopped playing the game in the first level left a negative review, which is an extreme number compared to the other games. The reviews from the level mention various bugs and broken storyline, issues that seem to have been mostly fixed in later patches. The last level also seems to contain problems, although fewer players that quit the game while passing through it wrote a negative review.

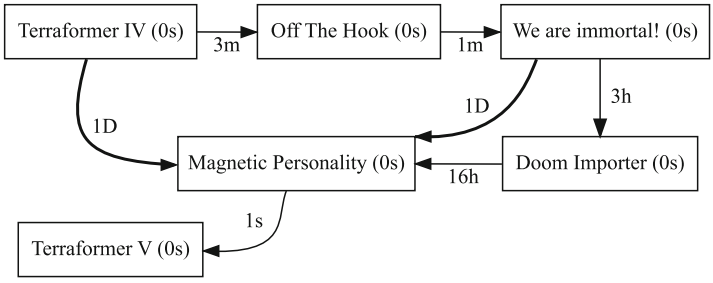

Fig. 4. Performance model of the last level of the game G4. Discovered as a directly-follows-graph with 80% of variants shown.

Focusing solely on the last level, the performance model in Fig. 4, the last bottleneck that the reviews mention corresponds to the achievement *Magnetic Personality*, which is a prerequisite of finishing the game and, based on the model, a clear bottleneck. All this points towards a problem that discouraged a significant number of players from finishing the game.

5.4 Noise Detection

Following analysis focusing on all anomalous behaviour, regardless of the cause being bugs or cheating players, makes the anomalies easier to detect. The aim is finding behaviour that does not represent the common player behaviour.

However, in cases of having the knowledge of the game and its achievements more deeply, we can use a similar technique to find behaviour that is not desired, i.e. caused by bugs or cheating. This would be done by constructing the reference process model by hand, not discovering it.

Table 2. Fitness of selected process models as discovered by the token replay algorithm.

	Fitness
Figure 1(a)	0.798
Figure 1(b)	0.782
Figure 1(c)	0.990
Figure 2 (left)	0.941
Figure 2 (right)	1

The conformance checking is utilized to find inconsistencies in the traces that might successfully filter out unwanted logged behaviour. In our case, first, the process model is discovered, using Heuristics miner algorithm, as it filters out infrequent behaviour [20]. As the token replay algorithm is utilized for conformance, the model must be a Petri net, but a heuristic net can be converted into one for this purpose. Then, the discovered Petri net and the original event log are put as an inputs to token replay algorithm, which outputs evaluation of fitness. Concretely, the values for the models in Figs. 1 and 2 are in Table 2.

To determine the accuracy of the token replay, we can compare its results with the manually found incorrect traces (described in Sect. 4.5). Ideally, we want to find a significant overlap between incorrect and unfit traces. Naturally, no such check can be done for games with no progress achievements.

Out of the total 1,002 traces, we have found 59 unfit traces through conformance checking, while 71 traces are classified as incorrect by manual filtering. Noticeably, out of the 59 unfit traces, 58 were classified as incorrect manually, leaving only one unfit trace that was not caught by the manual filtering. In the case of cheating traces, 5 out of the 6 are classified as unfit. Overall, the conformance checking discovered a large percentage (81.69%) of anomalous traces.

6 Discussion

We have shown how process mining techniques can be utilized in the context of video game analysis based on achievement data. In this setting, four use cases were shown. Table 3 shows which categories of games are appropriate for the specific use case.

The discovery of a process model that shows common player behaviour was demonstrated as not being limited by the game linearity. We have further shown that a variety of questions can be answered by examining the process model

Table 3. Appropriate types of games for specific analyses.

	Game linearity		
	Linear (P+0)	Slightly Linear (P+0)	Non-linear
Typical playthrough	✓	✓	✓
Comparing player behaviour (Un/finished)	✓	✓	✗
Game level analysis	✓	*?*	✗
Noise detection	✓	✓	*?*

with the aim of game improvement. For example, if there is a suspicion that specific activity in the game is causing problems, we can examine the activities and behaviour that players exhibited after passing it.

Furthermore, we showed usage of process mining for comparing behaviour of selected player groups. Concretely, players who did and did not finish a game. While this specific comparison is possible only for games with progress achievements, there is virtually no limitation on the type of players that can be compared. For example, players who have written a positive/negative review, players who did and did not unlock a particular achievement in the game, players who spent more/less time playing than a specified time limit. However, if the process model needs to contain all of the behaviour found in the event log, the dependency threshold should be set to -1.

Conformance checking, specifically token replay, was used to verify the appropriate fitness of the discovered process models. We applied token replay to both the linear and non-linear games. While the approach was sufficiently accurate for the linear games, there is not enough information about the non-linear games to determine the accuracy of the approach, mandating further research.

Lastly, we have shown that suggestion to level design can be based on the bottleneck analysis when the additional player statistics are also considered. The analysis is also limited to games that contain progress achievements.

7 Conclusion

This paper presented four use cases for process-focused game analytics, based on game achievement data. Within this setting, the usability of process mining techniques was demonstrated as a viable way to get much-needed insight for game design and beyond. Moreover, we categorized the games and showed their influence on the applicability of process mining.

All analyses were performed with a particular use case in mind, which we believe will ease possible adoption in the game industry. Specifically, player behaviour was observed in the process models, together with the demonstration of the possibility to compare between players groups. Furthermore, aiming at the purpose to improve level design, bottleneck analysis was performed, with the possibility of in-depth analysis. Taking the results into account, cross-check

with other player statistics was recommended. Lastly, conformance checking was utilized to discover faulty traces in an event log to identify bugs or cheating players. In summary, the applicability and utility of process mining in game design were clearly illustrated.

References

1. van der Aalst, W.: Process Mining: Data Science in Action, 2nd edn. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49851-4
2. Apperley, T., Gandolfi, E.: Evaluating gamer achievements to understand player behavior. In: Data Analytics Applications in Gaming and Entertainment (2019)
3. Baek, Y., Touati, A.: Exploring how individual traits influence enjoyment in a mobile learning game. Comput. Hum. Behav. **69**, 347–357 (2017)
4. Cruz, C., Hanus, M.D., Fox, J.: The need to achieve: players' perceptions and uses of extrinsic meta-game reward systems for video game consoles. Comput. Hum. Behav. **71**, 516–524 (2017)
5. El-Nasr, M., Drachen, A., Canossa, A.: Game Analytics: Maximizing the Value of Player Data. Springer, London (2016). https://doi.org/10.1007/978-1-4471-4769-5
6. Ghasemi, M., Amyot, D.: From event logs to goals: a systematic literature review of goal-oriented process mining. Requirements Eng. **25**(1), 67–93 (2019). https://doi.org/10.1007/s00766-018-00308-3
7. Groening, C., Binnewies, C.: "Achievement unlocked!" - the impact of digital achievements as a gamification element on motivation and performance. Comput. Hum. Behav. **97**, 151–166 (2019)
8. Heeter, C., Lee, Y.H., Medler, B., Magerko, B.: Conceptually meaningful metrics: inferring optimal challenge and mindset from gameplay, p. 36, March 2013
9. Jakobsson, M.: Achievement anatomy. In: Debugging Game History: A Critical Lexicon, p. 1 (2016)
10. Macak, M., Kruzelova, D., Chren, S., Buhnova, B.: Using process mining for git log analysis of projects in a software development course. Educ. Inf. Technol. **26**, 1–31 (2021)
11. Mans, R.S., Van der Aalst, W.M., Vanwersch, R.J.: Process Mining in Healthcare: Evaluating and Exploiting Operational Healthcare Processes. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16071-9
12. Montola, M., Stenros, J., Waern, A.: Pervasive Games: Theory and Design. CRC Press, Boca Raton (2009)
13. Myers, D., Suriadi, S., Radke, K., Foo, E.: Anomaly detection for industrial control systems using process mining. Comput. Secur. **78**, 103–125 (2018)
14. Nikitina, K.: Educational game analysis using intention and process mining. Master's thesis, National Research University Higher School of Economics (2020)
15. Prot, S., McDonald, K.A., Anderson, C.A., Gentile, D.A.: Video games: good, bad, or other? Pediatr. Clin. **59**, 647–658 (2012)
16. Ramadan, S., Ibrahim Baqapuri, H., Roecher, E., Mathiak, K.: Process mining of logged gaming behavior. In: 2019 International Conference on Process Mining (ICPM), pp. 57–64 (2019)
17. Shaker, N., Yannakakis, G.N., Togelius, J.: Digging deeper into platform game level design: session size and sequential features. In: Di Chio, C., et al. (eds.) EvoApplications 2012. LNCS, vol. 7248, pp. 275–284. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29178-4_28

18. Siqueira, E., Castanho, C., Rodrigues, G., Jacobi, R.: A data analysis of player in world of warcraft using game data mining. In: 2017 16th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames), pp. 1–9, November 2017
19. Schaedler Uhlmann, T., Alves Portela Santos, E., Mendes, L.A.: Process mining applied to player interaction and decision taking analysis in educational remote games. In: Auer, M.E., Langmann, R. (eds.) REV 2018. LNNS, vol. 47, pp. 425–434. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-95678-7_47
20. Weijters, A., van der Aalst, W.M., De Medeiros, A.A.: Process mining with the heuristics miner-algorithm. Technische Universiteit Eindhoven, Technical report WP 166, pp. 1–34 (2006)