StABLE: Analyzing Player Movement Similarity Using Text Mining

Luana Fragoso
Department of Computer Science
University of Saskatchewan
Saskatoon, Canada
luana.fragoso@usask.ca

Kevin G. Stanley
Department of Computer Science
University of Saskatchewan
Saskatoon, Canada
kstanley@cs.usask.ca

***Abstract*—Digital games are increasingly delivered as services. Understanding how and why players interact with games on an ongoing basis is important for maintenance. Logs of player activity offer a potentially rich window into how and why players interact with games, but can be difficult to render into actionable insights because of their size and complexity. In particular, understanding the sequential behaviour in-game logs can be difficult. In this paper, we present the String Analysis of Behaviour Log Elements (StABLE) method, which renders location and activity data from a game log file into a sequence of symbols which can be analyzed using techniques from text mining. We show that by intelligently designing sequences of features, it is possible to cluster players into groups corresponding to experience or motivation by analyzing a dataset containing Minecraft game logs. The findings demonstrate the validity of the proposed method, and illustrate its potential utility in mining readily available data to better understand player behaviour.**

***Index Terms*—log analysis, bag of words, movement, motivation, experience, data mining, data analytics**

I. Introduction

Interactive digital entertainment is part of life in the Information Age. In America, 64% of households own a device used to play games and 60% of Americans play video games daily [1]. All of this gaming activity is a significant economic driver. Video games moving into the mainstream has created additional pressure on developers to provide high-quality games on strict schedules, and to maintain them after launch to protect in-game monetary streams, which can pass 1 billion USD annually for games like Fortnite [2].

Improving game log analysis is an attractive target for developers interested in understanding how and why players interact with their games. Game logs are entirely under the control of the developers, can include substantial detail about player activities, and can unobtrusively collect data from actual play sessions. For games which simulate worlds, and movement through those worlds, the traces of where a player went, how they got there, and what they did while in transit could be mined for player behaviours.

Movement and actions through virtual worlds are the primary mechanics in open-world (sandbox) games. However, there is a lack of studies addressing analysis in open-world games with user-generated or procedure-generated content [3], [4]. This type of game is complex to analyze because of its freedom of letting players play as they want. There are no missions or pre-defined goals, not even an end. Sandbox games cannot directly benefit from most current player modeling methods as they are based on anchoring analysis on goals and achievements [5]–[7]. Few authors have addressed methods for analyzing player behaviours for sandbox games [8]–[11], but none based on movement analysis. Movement behaviour is extensively studied for real-world applications in GIScience, and can be applied to modeling virtual movements [12], [13].

In this paper, we present the String Analysis of Behaviour Log Elements (StABLE) method. We derived movement behaviours from location data based on previous work in GIScience [14]–[16]. Our method keys on the observation that location and action patterns through time can be represented as long strings, and therefore could be amenable to text mining techniques. We employ the canonical Bag of Words, N-gram, and cosine similarity tools [17] to compare different aspects of player movement behaviour. Depending on how the strings are encoded, different questions can be answered. We evaluate the utility of StABLE by using it to answer the following recurrent questions in the game user research (GUR) field:

RQ1: Do players have similar visited frequency behaviours in the game? [12], [18]

RQ2: Do players change their visited frequency behaviour through the game? [19], [20]

RQ3: Do players have similar movement behaviours when comparing themselves through the game? [5], [6], [21]

RQ4: Do players with similar expertise level move in the same manner through the game? [7], [22], [23]

RQ5: Can we cluster the players based on the self reported motivation given their movement similarities? [24], [25]

The findings from the RQs show that StABLE is versatile, efficient, and able to logically discern important play style and play motivation patterns amongst users. Because we are not design experts, the intention of addressing those questions is to demonstrate the flexibility of StABLE as a method to answer many different questions around gameplay rather than to make a specific contribution to game studies. For the sake of space, we do not compare StABLE to other methods in the results of this paper, and leave this study as a future work.

978-1-6654-3886-5/21/$31.00 ©2021 IEEE

II. Background and Related Literature

Player modeling based on movement behaviour has potential because movement is a primary mechanic in open-world games [12]. Movement analysis is not new in GUR, but is traditionally visualized as heatmaps or trajectories [18], [19], [26]. Recent work by Aung et al. [23] is the first to add spatio-temporal data metrics with the DEDICOM algorithm [20], [27], among other behavioural metrics, to classify players in open-world games. However, one must know the semantics of the world map to interpret the results, which can be impractical for procedurally generated games like Minecraft.

Movement analysis from GIScience include the use of features derived from absolute locations and time, called movement parameters (MPs) [28]. Visit frequency, dwell time, and trip duration are popular MPs found in many works [14], [15], [29], [30], and could be easily applied for movement analysis in virtual world [12]. Visit frequency is the number of times an individual visited a location. Dwell time is the duration an individual stayed in the same location. Trip duration is how long an individual moved between locations. Dodge et al. [15] used a variant of the Edit Distance (ED) algorithm to compare MPs of hurricanes. In GUR, Glyph [31] and Gamalyzer [22] also use a variant of ED to compare game states and goals.

Computing this matrix for long strings is prohibitive, and might require pruning techniques such as in Gamalyzer [22], where behaviours beyond a time window size k are not compared to each other, resulting in a computational complexity of $O(m \times k)$. This pruning works for linear games [22], [31] because behaviours that happen later on in the game are performed in a different context than those at the beginning of the game [22]. Popular and more efficient techniques in text mining are N-grams, Bag of Words (BoW), TF-IDF, and cosine similarity [17].

N-grams capture sequential patterns of length N by treating combinations of words as new words.

Bag of Words (BoW) combines two or more strings by taking the union of the observed words from each string. When BoW is combined with N-grams, the histogram can be built for a specific N-gram, or for a range of N-grams. For example, a BoWs built from trigrams would be a histogram of all three word sequences in the text, but a BoW built over a range $N = \{1, 2, 3\}$ would contain a histogram containing all single words, all two word sequences and all three word sequences in the text.

Term Frequency - Inverse Document Frequency (TF-IDF) normalizes the count of the word histograms to fairly compare strings with unequal length. Term Frequency (TF) is the ratio of the count of a word in a string divided by the number of words in the same string.

Cosine Similarity is a canonical equation to calculate similarity between vectors as the cosine angle between the vectors. The cosine similarity of two documents ranges from 0 to 1, where 0 means they are not equal and 1 means they are totally similar.

Lee et al. [21] reported a general approach to identify bots based on player self-similarity using cosine similarity. They created vectors to represent the amount of activity a player performed over a specific time frame. They employed cosine similarity to evaluate the self-similarity of players through the game.

A more general method able to analyze player behaviours, individually and in groups, that also works for more complex game analysis, like for sandbox games, is needed [3], [4]. Although Gamalyzer [22] and Glyph [31] showed potential use of ED to analyze sequence of actions, they are not a good fit for sandbox games because of their poor scaling for long strings, and strict ordering comparison. Similar to Lee et al. [21], we use a frequency vector and cosine similarity to compare behaviours, but we do not compare the vectors for each interval.

Popular data mining methods in GUR such as Hidden Markov Models (HMM) [5], Frequent Pattern Mining (FPM) [32], and Neural Networks (NN) [33] while effective, provide less observable and therefore descriptive outcomes. HMM strongly assume that a behaviour happens based on the previous one which is a poor model for non-linear sandbox games. Finding the right support value for FPM is challenging. While low support returns too many patterns that further analysis is impossible, high support returns a few patterns that might not be conclusive. NN require significant amount of data to achieve accurate and repeatable results. Collecting data from video games is not always easy, since the datasets are usually private [3].

III. StABLE

The StABLE concept is summarized in Figure 1. The method aggregates movements as MPs and encodes them as strings to use text mining techniques for comparison. Each step presents different possible implementations. In the following paragraphs, we explain the concept of each step and how we implemented them.[1]

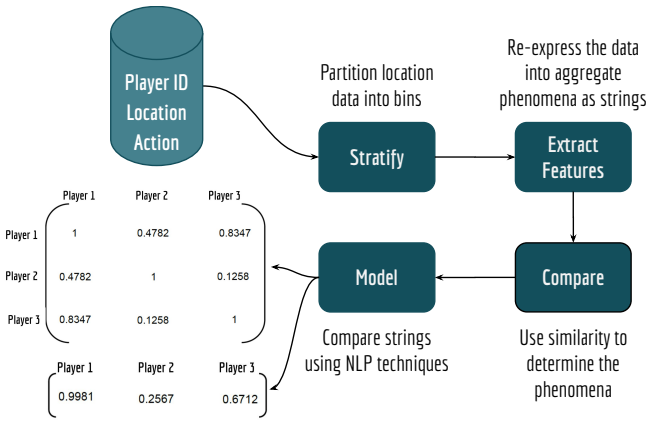

Fig. 1. StABLE concept

Given a preprocessed dataset ordered by timestamp containing the player ID, locations, and actions, StABLE first

[1]Our implementation in Python (v3.7) can be found at https://github.com/fragosoluana/stable-minecraft

sorts the locations into bins to simplify the detection of MPs (**stratification**). Our implementation requires a dataset with columns containing locations in four dimensions: x, y, z, and world label, because many video games have different worlds or levels. StABLE stratifies the locations into a grid by concatenating the locations of each position x, y, and z plus the world. For example, if the player was in world 0 and at the location $(217, 914, 493)$, the final cell label would be the string `0-217-914-493`. Cell size is a parameter. Discretized locations are stored in an intermediate variable `loc_bin`.

In the **feature extraction** step, player behaviours are encoded as strings. As player behaviour unfolds through time, a record of symbols representing that behaviour can be generated. The nature of the symbols selected determines the type of analysis conducted and by extension, questions answered. Changing these parameters changes the operationalization of the comparison. For example, comparing two strings based only in the location bins, the comparison will answer the question "*Does player A follow the same trajectories as player B?*". If we added the timestamp to the trajectory record the comparison would prefer co-located play. Because timestamp and location are individual words, players who visited the same locations at different times would be more similar to co-located players than players who never visited those location, as the Bag of Words (BoW) would treat location and time as independent words in the histogram. However, if location and time were merged into a single word `timestamp1-loc_bin1 timestamp2-loc_bin2...` co-located play would be required for similarity, as bins in the BoW histogram require both the time and location to be identical to be considered the same. To illustrate the power and flexibility of this technique we developed three different strings: the location, dwell-trip, and dwell-trip-action constructions.

Location construction. We chose to create the location construction to show that StABLE can be used for analysis based solely on visited locations. To probe use of space using StABLE, we encoded the location bins as separate words. Players who visited the same places, more often, would be more similar. The `loc_bin` attribute is enough to describe this behaviour.

Dwell-trip construction. To render strings which capture movement patterns independent of location, we aggregate movement into periods of motion (trips) and stationary periods (dwells). For every sequential record where player location is unchanged, dwell time is incremented (`X`). For every sequential record where player location is different, trip duration is incremented (`Y`), until the player remains at the same location for at least three timestamps, breaking the trip [14]. This aggregation naturally transforms the location records into pairs of dwells (`d`) and trips (`t`), and captures *how* players move, instead of *where*. Each player's trajectory can be expressed as a string of the form: $\texttt{dX}_1\ \texttt{tY}_1\ \texttt{dX}_2\ \texttt{tY}_2 \ldots$.

Dwell-trip-action construction. Designers might not only be interested in how players move, but what actions they undertake while moving (for example attacking, building, or interacting with others). We created a string incorporating whether or not an action had occurred during each dwelling or movement period. If at least one action has occurred, the word `a1` is added after the dwell time or trip duration, otherwise, `a0` is added. The resulting string is $\texttt{dX}_1\ \texttt{aZ}_1\ \texttt{tY}_1\ \texttt{aZ}_2\ \texttt{dX}_2\ \texttt{aZ}_3\ \texttt{tY}_2\ \texttt{aZ}_4 \ldots$, where `Z` can be 0 or 1.

Between-player and within-player analysis. Comparing players with each other can be used to segment players into clusters of playstyles, but comparing players with themselves can also be useful to describe how gameplay changes with time or through phases (tutorial, main game, end game) or in within player experiments (for example before and after a patch is deployed). We implemented StABLE to be able to compare different players (between-player analysis), and behaviours from a single player through time (within-player analysis). In the between-player analysis, all the data from each player is compared to each other. In a within-player perspective, StABLE performs a simple per-player median split on their play log to obtain two strings to be compared.

To **compare** the constructed strings, the BoW with TF-IDF is performed given an N-gram value. N-gram is related to the temporal aspect of the MPs, and subsequently how the similarity is interpreted. For the simple location string the N-gram is the sequence of movements, where $N = 1$ directly reproduces standard aggregate location distributions. The dwell and trip construction is the sequential movement patterns and naturally follows pairs of `dX tY`, which implies that N should increase by twos to capture subsequent motion. Similarly, N should increase by fours in the case of the dwell-trip-action construction. After constructing the BoW histogram, similarity can be calculated using cosine similarity.

The final step of StABLE (**model**) returns a matrix for a between-player analysis, and a simple list for a within-player analysis, as shown in Figure 1 with 3 players. The similarity matrix (sim) in a between-player analysis is a symmetric matrix $P \times P$, where P is the total number of players, with the diagonal populated by ones, where each element in the matrix sim_{ij} is the similarity score between the behaviours of players i and j. In a within-player analysis, the list contains P scores.

IV. Running StABLE

We collected data from the canonical sandbox game Minecraft [34] to answer the five questions in Section I. Minecraft worlds are procedurally generated with voxels that represent resources in the world. Minecraft is popular in the research community [35].

A. Study Procedure and Apparatus

We recruited 40 participants (7 female) aged 18 to 34 ($\mu = 20.97, \sigma = 3.63$). Participants already had a Minecraft account prior to participating in the study. Thirty six of the participants reported playing digital games at least a few times per week. Participants completed informed consent, and demographics questionnaires before playing. Enrollment happened via a website, and under the authorization of our institutional ethics review board. After completing the surveys,

participants were added to the server white-list and received an email on how to install Minecraft Forge in their computers and connect to the server. Participants were instructed to play normally. Data collection occurred over 2.5 weeks. At the end of the study, participants completed the SIMS test [36]. Participants received an honorarium of CAD $25.

We collected the game log using Minecraft Forge [37], a framework to facilitate the interface between Minecraft and player designed mods. We built a custom data logging mod that collects 25 different events from the game each second. At the end of the experiment, each event in the game log was uploaded to a table in a MySQL database. A Minecraft multi-player server was installed on a Virtual Machine running Linux (RHEL 6.10) with 2 CPUs and 4 GB of RAM. The server ran Minecraft Forge v13.20.0.2228 on Minecraft v1.11.2 using our mod. Participants accessed the server from their own personal computers, but they had to install the same Minecraft Forge version to avoid conflicts. All players started the game in the same position and played with the same randomly selected world seed (5791568963867681365).

B. Preprocessing

From the perspective of a designer using StABLE, the only information needed are the columns of the dataset representing player's location and action. Data were selected and filtered from Minecraft game logs to address the five research questions in Section I. We selected four tables from our database that include the necessary data to run StABLE. Cited columns were employed in our analysis.

PlayerTick stores time, player ID, position x, position y, position z, current dimension, experience level, among others. This dataset has 1.6 million records.

PlayerInteract provides data when the player is about to interact with a block. It has 5.1 million records of time, player ID, among others.

PlayerLoggedIn provides data when a player connects to the server. We recorded 466 records of time and player ID.

PlayerLoggedOut provides data when a player disconnects from the server. We recorded 448 records of time and player ID.

After selecting the necessary data, we filtered the dataset to remove unwanted data. Because we were not policing player activity while playing Minecraft, there was a chance of some of them used bots to advance in the game. With *PlayerLoggedIn* and *PlayerLoggedOut* tables, we could calculate how long players played per session to possibly eliminate these bots. For each *PlayerLoggedOut* data point, we subtracted its time to the closest time before in *PlayerLoggedIn* with the same player ID. We then took the average hour per session. Only two players played for longer than 8 hours on average, who were eliminated from the *PlayerTick* dataset as we suspected bots were playing.

Few players interacted with the game for more than 1.5 hours at a time. If players had dwell times in excess of 1.5 hours, the corresponding data was removed because we suspected that the game had been left open. Because we analyze player's motivation for RQ5, only the players in *PlayerTick* who answered the SIMS questionnaire were selected. At the end of the filtering, *PlayerTick* had a total of 30 players with 1.4 million records.

We aggregated both *PlayerInteract* and *PlayerTick* to the level of a second for each player. When aggregating, we took the mean location and the total number of actions. With single data points for each timestamp, we were able to merge both datasets based on their player ID and timestamp. The resulting dataset G (Table I) had a total of 30 participants and around 1.1 million records, and the following columns: player (*player_id*), location (*pos_x, pos_y, pos_z*, and *world*), and number of actions in a timestamp (*n_action*). The dataset G is given to StABLE, which generates the three string constructions and returns the between-player and within-player similarities.

TABLE I
FINAL SCHEMA OF GAME LOG G DATASET WITH AROUND 1.1 MILLION RECORDS. THIS DATASET IS USED TO RUN STABLE.

Attribute	Description
player_id	Player's ID
pos_x	Minecraft coordinate for position in x
pos_y	Minecraft coordinate for position in y
pos_z	Minecraft coordinate for position in z
world	Minecraft world where the player is located
n_action	Number of voxel interactions

In addition, we classified players as advanced and non-advanced to cluster them based on their expertise. Minecraft players can obtain experience levels [11]. There are three worlds in Minecraft: the Overworld, the Nether, and the End. All players start in the Overworld, where they need to open a portal if they want to reach the Nether. Players need to open a portal to reach the End with items found exclusively in the Nether. The Nether also contains unique resources that can be used to create potions, weapons and armor.

We retrieved the maximum experience level (MEL) ever reached for each player, which reflects the longest period they were alive, and their activity during that period. We then established an approximation for expertise where players with 21 or more MEL were labelled as advanced because that was the lowest MEL of any player who reached the Nether, resulting in 18 advanced players and 12 non-advanced players.

The SIMS scores and expertise information were merged in a dataset P that is used to interpret the model from StABLE. The final P dataset containing the player ID, expertise, and SIMS scores is shown in Table II, with 30 records and 6 attributes. With this dataset and the similarity matrix, the five research questions posed by this paper can be addressed.

V. ANALYZING STABLE SIMILARITIES

We investigated the similarities in a between and within-player analysis. We applied statistical tests in all of the five research questions to differentiate group of players, with the null hypothesis that the group of players (expertise or motivation) do not behaviour differently. We used the Mann-Whitney U statistical test because similarity-based partitioning

TABLE II
FINAL SCHEMA OF PLAYERS P DATASET WITH 30 RECORDS. SIMS SCORES WERE CALCULATED BASED ON [36]. THIS DATASET IS USED TO INTERPRET THE RESULTS FROM STABLE.

Attribute	Description
player_id	Player's ID
in_motivation	SIMS intrinsic motivation
id_regulation	SIMS identified regulation
ex_regulation	SIMS external regulation
amotivation	SIMS amotivation
xp	Advanced or non-advanced

does not necessarily return normal distributions. We reported the results with a significant effect when $p < 0.05$. We used the library *stats* under *scipy* (v 0.19.1) in Python.

We ran StABLE with the dataset G (Table I) using the default grid stratification with cells of size of 1, which is a voxel in Minecraft. For the sake of space, we only show figures of the results for a selection of N-grams. A Mac-Book Pro 2.7 GHz Intel Core i7 with 16 GB 1600 MHz DDR3 RAM was used to perform the analysis. The run time and memory use of StABLE for each research question did not exceed 2 minutes or 200MB respectively, not including the time and memory required to download and store the original data.

A. Location Analysis

The location construction was used to answer RQ1 and RQ2. The former requires the similarities between players. We represented the similarity matrix as a box plot in Figure 2, where each box plot is a row of the matrix. The highest similarity between two players was 76.89% ($\sigma = 14.38\%$), where only 7 pairs of players had a similarity over 50%. An analysis with $N = \{1, 2\}$ and $N = 2$ resulted in lower similarities with the highest similarity at 68.09%, $\mu = 8.17\%$, and $\sigma = 7.82\%$; and, the highest similarity at 43.64%, $\mu = 0.65\%$, and $\sigma = 3.64\%$, respectively. Because each player has a distribution of similarities with all other players, we took the mean of each row of the similarity matrix, and tested whether there were differences between non-advanced and advanced players in mean behavioural similarity. The independent variable was the player experience with 2 levels (advanced and non-advanced), and the dependent variable was the mean similarity distribution, which did not find a significant difference between the groups ($p > 0.05$) for any N value.

RQ2 is a within-player analysis, and we represented the scores as a bar plot (Figure 3). Players are more similar to themselves through time than with each other with $N = 1$. The highest similarity is 91.97%, $\mu = 57.88\%$, and $\sigma = 20.62\%$. An analysis with $N = \{1, 2\}$, and $N = 2$ resulted in lower similarities with the highest similarity at 89.5%, $\mu = 52.09\%$, and $\sigma = 20.66\%$; and, the highest similarity at 74.85%, $\mu = 29.6\%$, and $\sigma = 22.28\%$, respectively. The statistical test showed that player expertise was not distinguishable to a level of significance ($p > 0.05$) for any N value, where the

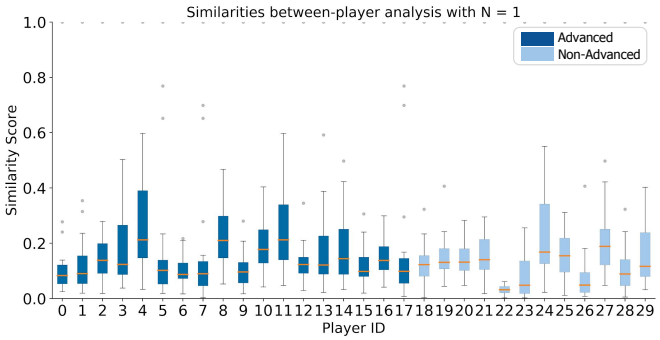

Fig. 2. Similarity scores between players using the location construction to answer RQ1. Players are in descending order of experience level in the x-axis. The lower and upper quartile values of the similarities (box), the median (line), and the range of the similarities (whiskers) are illustrated.

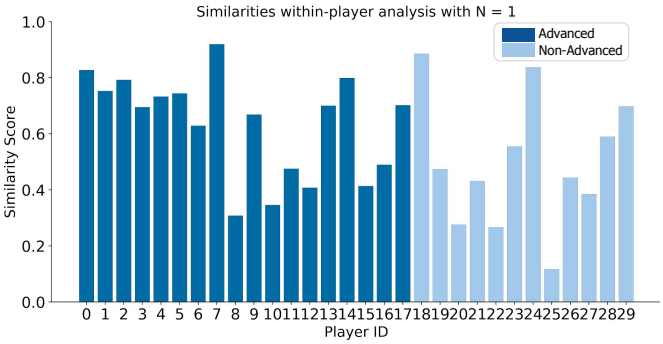

Fig. 3. Similarity scores in a within-player analysis using the location construction to answer RQ2. Players are in descending order of experience level in the x-axis.

independent variable was player experience (2 levels), and the dependent variable was the similarity scores.

B. Playstyle stability

For RQ3, we used the dwell-trip construction to investigate playstyle stability in a within-player analysis. We visualize the scores with a bar plot. In Figure 5 ($N = 2$), 77% of the advanced players had trip behaviour similarity over 50%. An $N = \{1, 2, 3, 4\}$ showed even higher similarity scores for advanced players. Most non-advanced players showed a similarity closed to 60%. Doubling the N-gram to 4 resulted in small similarities. We applied statistical tests to verify if advanced players had more stable playstyles (more similar to themselves) than non-advanced players. The test showed different sensitivity to playstyle changes for $N = 2$ ($p < 0.001$), $N = 4$ ($p < 0.05$), and $N = \{1, 2, 3, 4\}$ ($p < 0.0005$), with player experience as the independent variable (2 levels) and similarity scores as the dependent variable.

C. Player similarity with experience

We investigated RQ4 with the dwell-trip construction in a between-player analysis. Figure 4.a shows the similarity distributions among players with $N = 2$. The similarities ranged from 10.15% to 92.17%($\mu = 52.44\%$, $\sigma = 17.08\%$). Advanced players showed an overall similarity average higher

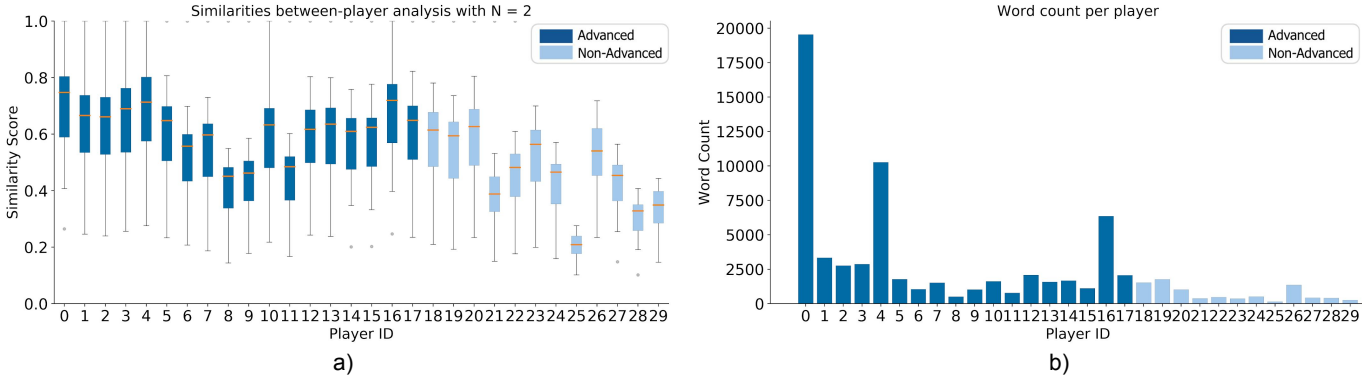

Fig. 4. Players are in descending order of experience level in the x-axis. (a) Similarity scores between players using the dwell-trip construction to answer RQ4. (b) Total number of words per player with the dwell-trip construction.

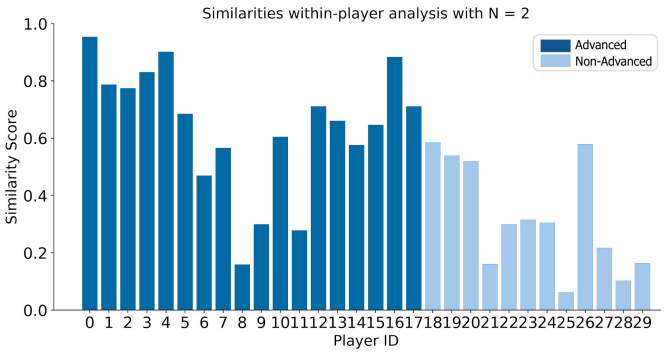

Fig. 5. Similarity scores in a within-player analysis using the dwell-trip construction to answer RQ3. Players are in descending order of experience level in the x-axis.

than non-advanced players, 59.39% against 45.97%. An $N = \{1, 2, 3, 4\}$ resulted in similarities ranging from 22.18% to 95.34% ($\mu = 65.58\%$, $\sigma = 15.03\%$). Advanced players still showed a higher similarity average (71.90%) than non-advanced players (58.99%). Similarities were lower with $N = 4$, ranging from 0% to 17.07% ($\mu = 1.98$, $\sigma = 1.94\%$). Advanced and non-advanced players had significantly different mean similarity scores ($p < 0.005$) for any N value. Note that StABLE is not strongly influenced by the length of the strings (Figure 4.b). For example, even though player 0 had a string with a length more than the double of the next largest number of words, his/her similarity was equivalent to others (3, 4, 5 and 17).

With the same dwell-trip construction, a network graph was employed as another way to visualize player similarity relationships for RQ4. Figure 6 shows the network graph for players who had a similarity of greater than 50%, with $N = 2$, and with connection to at least another player. Advanced players were similar to each other, but non-advanced players were more likely to show similar behaviours to advanced players than to the other non-advanced players. We can also depict from the graph that the most experienced player (player 0, with an experience level of 68) as well as player 4 (experience level of 35) presented the highest degree in the graph, both with 26 edges. All the advanced players appeared in the graph, whereas non-advanced players 25, 28, and 29 were not in the graph, who showed the lowest similarity distributions in Figure 4.a.

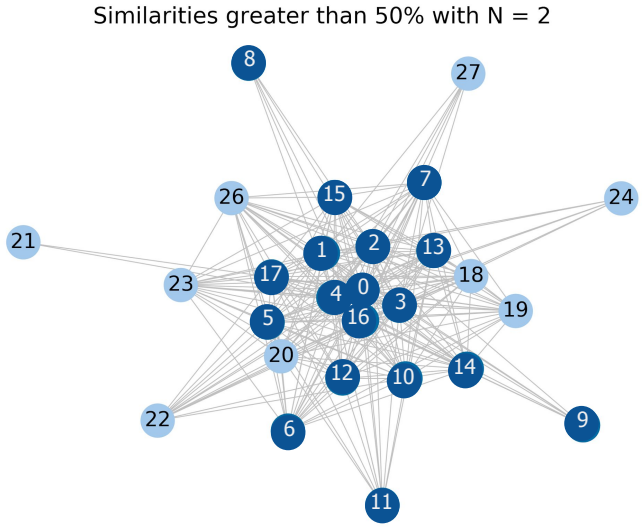

Fig. 6. Graph network in a between-player analysis using the dwell-trip construction for RQ4. The shorter the edges between the players, the more similar they are.

D. Player similarity and motivation

We used the dwell-trip-action construction for RQ5. We applied the Louvain community detection to cluster players based on their similarities with $N = 4$. We ran Louvain with predefined community labels to fix the number of resulting communities to avoid clustering into insignificantly small communities. We used average edge weight as the node importance. We performed a median split of average edge weight to seed communities. The start node for the Louvain method was the node with highest weight average in the network.

A graph with all the players resulted in a single community. Filtering out the lowest scores ($< 10\%$), resulted in two communities. The resulting network graph can be seen in Figure 7. We mapped the SIMS scores to the groups identified through Louvain, and applied the Mann-Whitney U test. Intrinsic motivation showed a significant difference across the player clusters ($p < 0.05$).

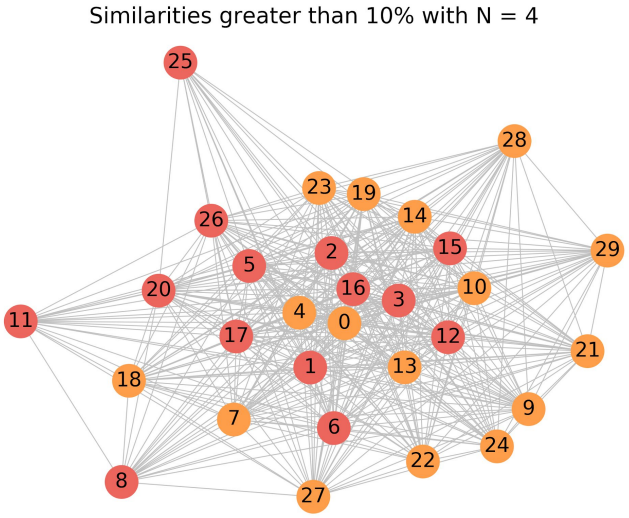

Fig. 7. Network graph of players in a between-player analysis with the dwell-trip-action construction, where each color shows a community as determined by the Louvain method. The closer the nodes, the more similar they are.

VI. Discussion

Our results demonstrate that it is possible to use StABLE to compare strings of movement and activity patterns to answer a number of important questions in GUR. We were able to compare the similarity of locations and trajectories both between and within-players. We found that players did not have similar location strings to each other nor with themselves, which might be due to Minecraft's mechanics. The expertise analysis showed that the movement and activity patterns associated with advanced players were more likely to be similar than non-advanced players. By aggregating away location, and examining patterns of travel and dwelling, we demonstrated that we could probe how people play. Unsupervised clustering of the similarities partitioned players by motivation, as encoded by the SIMS response, demonstrating that our technique can be used to identify why people play from how people play.

The findings we described here have strong internal validity, but weaker generalizability given the single game, limited demographic, and relatively small number of participants. However, the intent of this paper was not to provide strongly generalizable findings relating movement patterns to expertise, but to demonstrate that StABLE, as a tool for game log analysis, has substantial utility. The tool itself, and more importantly the concept underlying the tool, has broad applicability and can be used to analyze many types of sequential log data. While we focused on movement data in an open world game in this work, many other genres of games and streams of data, and player attributes could be analyzed.

Because we focused on providing broad evidence of applicability rather than focusing the application of StABLE to a single problem, this work provides more questions than answers with respect to game behaviours. We established that movement pattern similarity could cluster players a way that reflected their SIMS scores, but did not dig into which playstyle patterns were diagnostic. We demonstrated that, for Minecraft at least, advanced players had greater movement pattern similarity to each other than to non-advanced players, but did not probe the nature of those movement patterns. We showed that advanced players had more stable movement patterns than non-advanced players, but did not attempt to describe those patterns. Answering game play questions raised by StABLE is a potentially fruitful and fascinating avenue for future research.

The primarily limitation of this work is the scope of the data analyzed. By only analyzing a relatively small population over a single game, we cannot make strong conclusions about our game-related findings. As a methodological paper, this proof is sufficient; however, more detailed analysis of larger datasets, preferably obtained from actual commercial games is an obvious next step. Within the method itself, many potential avenues for further improvement exist. We employed the simplest (and most canonical) string comparison method. Extending this work to include more advanced and nuanced string comparison algorithms could be fruitful. The core contribution and insight in this work is the use of feature sequences to selectively probe specific behaviours. We have only explored three simple feature constructions: location, dwell-time and dwell-time-action. Many more sophisticated feature strings could be envisioned, and should be investigated.

VII. Conclusion

Game logs contain rich information on player experience, but this data can be difficult to mine for insights. We have presented the StABLE method for log analysis, which leverages techniques from text mining to perform comparisons of sequence of behavioural features. Using StABLE we were able to mine insights about player behaviour, expertise and motivation from Minecraft play logs collected over a two week period. The findings demonstrate the utility of our approach in game log analytics, and point the way towards both additional research in methods, and application to game log analysis in both commercial and research settings.

References

[1] E. T. E. S. Association, "Industry Facts," 2018. [Online]. Available: http://www.theesa.com/about-esa/essential-facts-computer-video-game-industry/

[2] Kellie Ell, "Video game industry is booming with continued revenue," 2018. [Online]. Available: https://www.cnbc.com/2018/07/18/video-game-industry-is-booming-with-continued-revenue.html

[3] D. Hooshyar, M. Yousefi, and H. Lim, "Data-driven approaches to game player modeling: A systematic literature review," *ACM Comput. Surv.*, vol. 50, no. 6, Jan. 2018.

[4] G. N. Yannakakis, P. Spronck, D. Loiacono, and E. André, "Player Modeling," *Artificial and computational intelligence in games*, vol. 6, pp. 45–59, 2013.

[5] S. Bunian, A. Canossa, R. Colvin, and M. S. El-Nasr, "Modeling Individual Differences in Game Behavior using HMM," *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2017.

[6] J. Pfau, J. D. Smeddinck, and R. Malaka, "Towards deep player behavior models in mmorpgs," in *Proceedings of the 2018 Annual Symposium on Computer-Human Interaction in Play*, ser. CHI PLAY '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 381–392.

[7] S. Ahmad, A. Bryant, E. Kleinman, Z. Teng, T.-H. D. Nguyen, and M. Seif El-Nasr, "Modeling individual and team behavior through spatio-temporal analysis," in *Proceedings of the Annual Symposium on Computer-Human Interaction in Play*, ser. CHI PLAY '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 601–612.

[8] S. Müller, S. Frey, M. Kapadia, S. Klingler, R. P. Mann, B. Solenthaler, R. W. Sumner, and M. Gross, "Heapcraft: Quantifying and predicting collaboration in minecraft," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2015.

[9] S. Müller, E. Zürich, M. Kapadia, S. Frey, S. Klingler, R. P. Mann, B. Solenthaler, and R. W. Sumner, "HeapCraft Social Tools: Understanding and Improving Player Collaboration in Minecraft," in *Proceedings of the 10th International Conference on the Foundations of Digital Games (FDG '15)*, 2015.

[10] S. Mueller, B. Solenthaler, M. Kapadia, S. Frey, S. Klingler, R. P. Mann, R. W. Sumner, and M. Gross, "Heapcraft: Interactive data exploration and visualization tools for understanding and influencing player behavior in minecraft," in *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games*, ser. MIG '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 237–241.

[11] A. Canossa, J. B. Martinez, and J. Togelius, "Give me a reason to dig minecraft and psychology of motivation," in *2013 IEEE Conference on Computational Inteligence in Games (CIG)*, 2013, pp. 1–8.

[12] M. S. El-Nasr, A. Drachen, and A. Canossa, *Game Analytics*. Springer, London, 2013.

[13] A. Drachen and A. Canossa, "Analyzing spatial user behavior in computer games using geographic information systems," in *Proceedings of the 13th international MindTrek conference: Everyday life in the ubiquitous era*, 2009, pp. 182–189.

[14] T. Paul, K. Stanley, N. Osgood, S. Bell, and N. Muhajarine, "Scaling behavior of human mobility distributions," in *Geographic Information Science*, J. A. Miller, D. O'Sullivan, and N. Wiegand, Eds. Cham: Springer International Publishing, 2016, pp. 145–159.

[15] S. Dodge, P. Laube, and R. Weibel, "Movement similarity assessment using symbolic representation of trajectories," *International Journal of Geographical Information Science*, vol. 26, no. 9, pp. 1563–1588, 2012.

[16] K. Farrahi and D. Gatica-Perez, "Discovering routines from large-scale human locations using probabilistic topic models," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 1, Jan. 2011.

[17] C. C. Aggarwal and C. Zhai, *Mining Text Data*. Springer, 2012.

[18] A. Canossa, T.-H. D. Nguyen, and M. Seif El-Nasr, "G-Player: Exploratory Visual Analytics for Accessible Knowledge Discovery," in *Proceedings of the First International Joint Conference of DiGRA and FDG*, 2016.

[19] G. Wallner, N. Halabi, and P. Mirza-Babaei, "Aggregated visualization of playtesting data," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, ser. CHI '19. New York, NY, USA: Association for Computing Machinery, 2019.

[20] C. Bauckhage, R. Sifa, A. Drachen, C. Thurau, and F. Hadiji, "Beyond heatmaps: Spatio-temporal clustering using behavior-based partitioning of game levels," in *2014 IEEE Conference on Computational Intelligence and Games*, 2014, pp. 1–8.

[21] E. Lee, J. Woo, H. Kim, A. Mohaisen, and H. K. Kim, "You are a Game Bot!: Uncovering Game Bots in MMORPGs via Self-similarity in the Wild," in *Proc. Netw. Distrib. Syst. Secur. Symp. (NDSS)*, 2016.

[22] J. C. Osborn and M. Mateas, "A Game-Independent Play Trace Dissimilarity Metric Categories and Subject Descriptors," in *Proceedings of the 9th International Conference on the Foundations of Digital Games (FDG '14)*, 2014.

[23] M. Aung, S. Demediuk, Y. Sun, Y. Tu, Y. Ang, S. Nekkanti, S. Raghav, D. Klabjan, R. Sifa, and A. Drachen, "The trails of just cause 2: Spatio-temporal player profiling in open-world games," in *Proceedings of the 14th International Conference on the Foundations of Digital Games*, ser. FDG '19. New York, NY, USA: Association for Computing Machinery, 2019.

[24] A. Canossa, "Give me a reason to dig: Qualitative associations between player behavior in minecraft and life motives," in *Proceedings of the International Conference on the Foundations of Digital Games*, ser. FDG '12. New York, NY, USA: Association for Computing Machinery, 2012, p. 282–283.

[25] N. Yee, "Motivations for Play in Online Games," *Cyberpsychology and Behavior*, vol. 9, no. 6, pp. 772–775, 2006.

[26] J. C. Campbell, J. Tremblay, and C. Verbrugge, "Clustering Player Paths," in *Proceedings of the 10th International Conference on the Foundations of Digital Games (FDG '15)*, 2015.

[27] R. Sifa, S. Srikanth, A. Drachen, C. Ojeda, and C. Bauckhage, "Predicting Retention in Sandbox Games with Tensor Factorization-based Representation Learning," *IEEE Conference on Computatonal Intelligence and Games, CIG*, vol. 0, pp. 1–8, 2016.

[28] S. Dodge, R. Weibel, and A.-K. Lautenschütz, "Towards a taxonomy of movement patterns," *Information Visualization*, vol. 7, no. 3-4, pp. 240–252, 2008.

[29] C. Song, T. Koren, P. Wang, and A. L. Barabási, "Modelling the scaling properties of human mobility," *Nature Physics*, vol. 6, pp. 818–823, 2010.

[30] R. Zhang, K. G. Stanley, D. Fuller, and S. Bell, "Differentiating population spatial behavior using representative features of geospatial mobility (refgem)," *ACM Trans. Spatial Algorithms Syst.*, vol. 6, no. 1, Feb. 2020.

[31] T.-H. Nguyen, M. Seif El-Nasr, T.-H. D. Nguyen, and A. Canossa, "Glyph: Visualization Tool for Understanding Problem Solving Strategies in Puzzle Games," in *Proceedings of the 10th International Conference on the Foundations of Digital Games (FDG '15)*, 2015.

[32] Z. Chen, M. S. El-nasr, A. Canossa, J. Badler, S. Tignor, and R. Colvin, "Modeling Individual Differences through Frequent Pattern Mining on Role-Playing Game Actions," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2015, pp. 2–7.

[33] J. Pfau, J. D. Smeddinck, and R. Malaka, "Deep player behavior models: Evaluating a novel take on dynamic difficulty adjustment," in *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, ser. CHI EA '19. New York, NY, USA: Association for Computing Machinery, 2019.

[34] Mojang, "*Minecraft*," Game [PC], Stockholm, Sweden, May 2009, mojang, Stockholm, Sweden. Last played March 2020.

[35] S. Nebel, S. Schneider, and G. D. Rey, "Mining learning and crafting scientific experiments: A literature review on the use of Minecraft in education and research," *Educational Technology and Society*, vol. 19, no. 2, pp. 355–366, 2016.

[36] F. Guay, R. J. Vallerand, and C. Blanchard, "On the Assessment of Situational Intrinsic and Extrinsic Motivation: The Situational Motivation Scale (SIMS)," *Motivation and Emotion*, vol. 24, no. 3, pp. 175–213, 2000. [Online]. Available: https://doi.org/10.1023/A:1005614228250

[37] F. Development LLC, "Minecraft forge," 2018. [Online]. Available: https://files.minecraftforge.net/