Proc. of the 2010 International Conference on Data Mining (DMIN'10), pp. 78-82,
July 12-15, Las Vegas, USA, 2010.

Detection of MMORPG Misconducts Based on Action Frequencies, Types and Time-Intervals

Ruck Thawonmas and Yoshitaka Kashifuji
Intelligent Computer Entertainment Laboratory
Graduate School of Science and Engineering, Ritsumeikan University
ruck @ ci . ritsumei . ac . jp

Abstract – *This paper describes an application of data mining to detecting misconducts, such as use of bots and cheats, in Massively Multiplayer Online Role-Playing Games (MMORPGs). A method is proposed that exploits the information on action frequencies, types, and intervals available in the targeted MMORPG log data. The aforementioned information is used as the input to a support vector machine. Evaluation results, using log data, available in rollback database, from Cabal Online, confirm the effectiveness of the proposed method.*

Keywords: Bot Detection, Cheat Detection, MMORPG, SVM, Cabal Online

1. Introduction

Misconducts such as bots and cheats in Massively Multiplayer Online Role Playing Games (MMORPGs) cause not only frustration to bona fide players but also financial loss to MMORPG publishers. MMORPG bots are tools for automatically creating items, fighting with monsters by setting, for example, mouse or character movement patterns. Cheats in MMORPGS refer to tools able to achieve performances beyond those designed or permitted by game publishers, such as moving through walls or being invulnerable during fighting. Detection of them is a challenging task for game publishers as well as researchers.

Recently, in order to cope with the increasingly MMORPG misconducts, researchers have proposed various techniques for preventing such misconducts with CAPTCHA [1, 2] or detecting them at the game level based on movement patterns [3-8], operating-system level based on window events [9], and traffic level [10]. However, they all have limited practical uses. A primary reason for practical use limitation is due to possible countermeasures by bot or cheat developers to evade the detection methods. We believe that detection algorithms that rely on more fundamental behaviors will be more robust against such countermeasures because any countermeasure in mimicking bona fide players' behaviors leads to low competency in fulfilling mala fide players' needs.

In our approach, we deal with MMORPG misconducts with a completely different perspective addressing their resource gathering and trading behaviors, the information of which is available in rollback database commonly maintained by game publishers. As resource accumulation is considered one of the most important goals among MMORPG players, misconduct tools must collect loots and trade them with non-player characters or other player characters. This can serve as a basis to detect

Figure 1 Screenshot of Cabal Online

misconducts in MMORPGs.

In this paper, we propose a method for detection of MMORPG misconducts that improves our recent method [11]. In addition to action frequencies and types used also in [11], the information on action intervals is exploited as the input to a support vector machine (SVM) [12]. Evaluation of the proposed method using rollback database from Cabal Online (Fig. 1) [13], published in Japan by Gamepot Inc., reveals 90% of the Matthews correlation coefficient (a typical quality measure for binary classification) [14] with the detection time (the amount of time to let a character of interest play for detection of its misconducts) of 60 mins, an 8% increase compared to our previous method [11].

2. Methodology

Following a brief description of our previous method in [11], we describe two methods. The former serves as an intermediate method exploiting action frequencies and types in a different fashion from the method in [11]. The latter is the proposed method where the information on action intervals is additionally exploited.

2.1 Our Previous Method

This method consists of two detection stages. The first stage detects misconducts by using only action frequencies. All characters not judged as irregular characters, those performing misconducts, are forwarded to the second stage that uses SVM to judge whether they are irregular characters.

At the first stage, character c is judged as an irregular character if at least one of its actions exists that satisfies all three conditions below:

$$\text{freq}(a,c) > \text{mean\_freq}(c) \quad (1)$$

$$\text{freq}(a,c) > \rho\,\text{max\_regular\_freq}(a) \quad (2)$$

$$\text{max\_regular\_freq}(a) > 0 \quad (3)$$

where freq(x,y), mean_freq(y), and max_regular_freq(x) denote the frequency of action x by character y, the mean of action frequencies by character y, and the action frequency of the regular player, the one with misconducts unobserved, in the training data who most frequently invoked action x, respectively. In addition, ρ is a parameter obtained from the training data through the following two steps:

- For each irregular character b in the training data, add action a to list L if its ratio $\frac{\text{freq}(a,b)}{\text{max\_regular\_freq}(a)}$ is the minimum among all actions that satisfy

$$\text{freq}(a,b) > mean\_freq(b) \quad (4)$$

$$\frac{\text{freq}(a,b)}{\text{max\_regular\_freq}(a)} \geq 1 \quad (5)$$

$$\text{max\_regular\_freq}(a) > 0 \quad (6)$$

- If L is not null, set ρ to the median of the elements of L; otherwise, set ρ to an extremely large value in order not to judge any character as irregular.

At the second stage, each of the characters not judged as irregular at the first stage is represented by a vector of m dimensions, where m is the number of action types. An element of the vector of a character of interest corresponds to one action type and is 1 when the character conducted the action and 0, otherwise. These m-dimensional vectors are used for training and testing SVM.

2.2 Intermediate Method

The second stage in the above method is not robust against countermeasures by bot or cheat developers. For example, they might develop misconduct tools that execute a wide range of actions to mimic regular characters, making the tools harder to be detected. All conditions employed in the first stage, (1)-(6), are heuristics and might lack generalizability for other MMORPGS.

An intermediate method for circumventing the above issues uses only SVM but with the m-dimensional input vector that takes into account not only action types but also action frequencies. Each element of the input vector has a value from 0 to 1, and the corresponding element for action a of character c is defined as follows:

$$\sqrt[4]{\frac{\text{freq}(a,c)}{\text{max\_freq}(c)}}, \tag{7}$$

where max_freq(c) denotes the frequency of the action that character c most frequently invoked and is used for normalizing the value of each vector element. In addition, the 4-th root is used for maintaining the influences of actions with less frequency by increasing their values; we have empirically found that this transformation gives promising results.

2.3 Proposed Method

The proposed method additionally takes into account the distribution of action intervals. This makes it harder for bot or cheat developers to perform counter measures against the proposed method because mimicking the action interval distributions of regular players will inevitably lower bot or cheat performances. As in the intermediate method, SVM is used. However, in addition to the aforementioned m elements, the input vector of SVM has t+1 more elements. Each new element indicating the frequency of the interval between two consecutive actions lying in [0, 1), [1, 2), ..., [t-1, t), and [t, ∞), respectively.

The m+i th element for the now m+t+1 dimensional vector of character c is defined as follows:

$$\sqrt[4]{\frac{\text{interval\_freq}(i,c)}{\text{max\_interval\_freq}(c)}}, \tag{8}$$

where interval_freq(i,c) denotes the frequency value of [i-1, i) for character c and max_interval_freq(c) the highest frequency among all t+1 intervals of character c. Note that the normalization and use of the 4-th root are used for the same purpose as (7).

3. Evaluation and discussions

The same Cabal Online data sets used in [11] are also used here for comparing the above three methods. Therein, four data sets were obtained by partitioning, with the overlap ratio of 0.5, the three-day log of each character into multiple chunks with the widths of 60, 45, 30, and 15 mins, respectively. Note that the chunk width indicates the amount of time required to let a character of interest play in order to detect its misconducts, henceforth called detection time. Each data set consists of seven regular characters of the Blader character class and seven detected irregular characters of the same class, where character levels are evenly distributed between regular and irregular characters.

A SVM tool called SVM-Light Support Vector Machine [15] with the linear kernel was used for obtaining the Matthews correlation coefficient (MCC)

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{9}$$

of each method for all four data sets, where TP is the number of true positives, TN the number of true negatives, FP the number of false positives and FN the number of false negatives. The parameters TP, TN, FP, and FN of each method for a data set of interest were

obtained by using cross validation. In each fold of a given data set, the test data consist of all chunks of a pair of one regular character and one irregular character, and the train data consist of all chunks of the remaining six regular characters and six irregular characters. Due to the number of possible combinations of one regular character and one irregular character for testing, there are 49 folds, leading to 49-fold cross validation.

Figures 2-5 show the MMCs of the three methods for the data set with the chunk width (the detection time) of 60, 45, 30, and 15 mins, respectively. For the datasets with the detection times of 60 and 45 mins, the MMC of the proposed method reaches 0.9 with sufficiently large t and outperforms the other two methods for most values of t. For the data sets with the detection times of 30 and 15 mins, the highest MMC of the proposed method is 0.84 and 0.76, respectively. For these two data sets, although the proposed method outperforms the other two methods for most values of t, the previous method has the best performance for the latter data set. This result poses a remaining challenging issue in achieving a high MMC with a short detection time, say 15 mins.

4. Conclusions

The proposed method using action frequencies, types, and intervals as the input to SVM with the linear kernel outperforms the other two methods in 2.1 and 2.2 for most cases in terms of the MCC. In addition, use of three different pieces of action information makes the proposed method robust against countermeasures by bot or cheat developers. A challenging remaining issue is to shorten the effective detection time.

Acknowledgement

The authors are much indebted to Hiroshi Kobayashi at Gamepot Inc, who helped us gather the game log. This work was supported in part by Research Grant for Promoting Technological Seeds (A), No. 1056, by Japan Science and Technology Agency and Grant-in-Aid for Scientific Research (C), No. 20500146, by Japan Society for Promotion of Science.

References

[1] P. Golle and N. Ducheneaut. Preventing bots from playing online games. *Computers in Entertainment*, 3(3): pp. 1-10, 2005.

[2] L. von Ahn, M. Blum, N. J. Hopper, and J. Langford. CAPTCHA: Using hard AI problems for security. In *Proc. of Eurocrypt*, pp. 294-311, 2003.

[3] C. Thurau, C. Bauckhage, and G. Sagerer. Learning human-like movement behavior for computer games. In *Proc. of 8th Int. Conf. on the Simulation of Adaptive Behavior* (SAB'04), pp. 315-323, 2004.

[4] C. Thurau and C. Bauckhage. Tactical waypoint maps: Towards imitating tactics in fps games. In *Proc. of 3rd International Game Design and Technology Workshop and Conference* (GDTW'05), M. Merabti, N. Lee, and M. Overmars, editors, pp. 140-144, 2005.

[5] K.-T. Chen, A. Liao, H.-K. K. Pao, H.-H. Chu. Game Bot Detection Based on Avatar Trajetory. In *Lecture Notes In Computer Science*, vol. 5309 (Proc. of 7th International Conference on Entertainment Computing), pp. 94 -105, 2008.

[6] K.-T. Chen, H.-K. K. Pao, and H.-C. Chang. Game Bot Identification based on Manifold Learning. In *Proc. of ACM NetGames 2008*, pp. 21-26, 2008.

[7] R. Thawonmas, J. Oda and K.-T. Chen. Analysis of User Trajectories Based on Data Distribution and State Transition: a Case Study with a Massively Multiplayer Online Game Angel Love Online. In *Proc. of 10th International Conference on Intelligent Games and Simulation (GAME-ON 2009)*, pp. 56-60, 2009.

[8] H. Kim, S. Hong, and J. Kim. Detection of auto

programs for MMORPGs. In *Proc. of AI 2005: Advances in Artificial Intelligence*, pp. 1281-1284, 2005.

[9] S. Mitterhofer , C. Kruegel , E. Kirda , C. Platzer. Server-Side Bot Detection in Massively Multiplayer Online Games. *IEEE Security and Privacy*, 7(3): pp. 29-36, 2009.

[10] K.-T. Chen, J.-W. Jiang, P. Huang, H.-H. Chu, C.-L. Lei, and W.-C. Chen. Identifying MMORPG bots: A traffic analysis approach. In *Proc. of ACM SIGCHI ACE'06*, 2006.

[11] R. Thawonmas, Y. Kashifuji, K.-T. Chen. Detection of MMORPG Bots Based on Behavior Analysis. In *Proc. of 2008 International Conference on Advances in Computer Entertainment Technology*, pp. 91-94, 2008.

[12] B. Scholkopf and A.J. Smola. Learning with Kernels. MIT Press, Cambridge, MA, 2002.

[13] Cabal Online
www.cabal.jp

[14] Matthews correlation coefficient
http://en.wikipedia.org/wiki/Matthews_correlation_coefficient

[15] SVM-LIGHT
svmlight.joachims.org

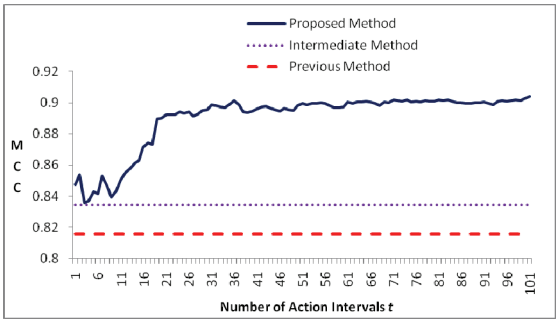

Figure 2 MMCs of the three methods for the data set with the detection time of 60 mins

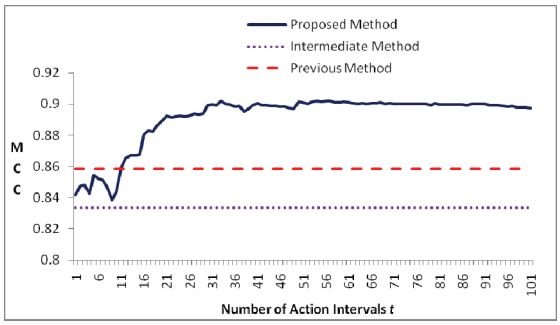

Figure 3 MMCs of the three methods for the data set with the detection time of 45 mins

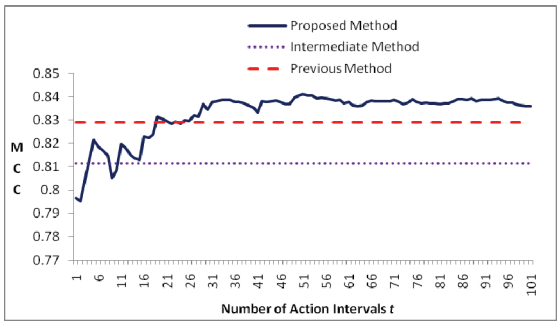

Figure 4 MMCs of the three methods for the data set with the detection time of 30 mins

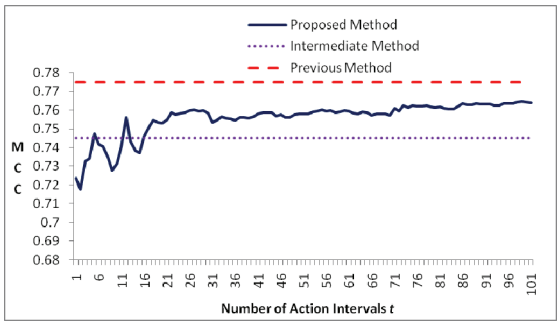

Figure 5 MMCs of the three methods for the data set with the detection time of 15 mins