Towards Reinterpretation of Interaction Complexity for Load Prediction in Cloud-based MMORPGs

Mirko Suznjevic
University of Zagreb
Faculty of Electrical Engineering and Computing
Unska 3, Zagreb, Croatia
E-mail: mirko.suznjevic@fer.hr

Maja Matijasevic
University of Zagreb
Faculty of Electrical Engineering and Computing
Unska 3, Zagreb, Croatia
E-mail: maja.matijasevic@fer.hr

***Abstract*—We present an approach for predicting the load of Massively Multiplayer Online Role-Playing Games (MMORPGs) in a cloud-hosted environment. The load of a MMORPG is determined not only by the number of players, but also by what they do in the virtual world. In our previous work, we have shown that player activity patterns can be categorized, and that each category may be represented by a corresponding mathematical model. In this work we extend the concept of *interaction complexity* applied for load prediction by Nae et. al. by taking into account MMORPG player behavior categories, and asses the prediction accuracy of the proposed approach.**

I. Introduction

Hosting Massively Multiplayer Online Role-Playing Games (MMORPGs) in a cloud-based environment offers the necessary resource flexibility to deal with highly fluctuating requirements of MMORPGs. The transition to cloud, however, opens a new problem, i.e., how to accurately predict the load of an MMORPG. In general, load estimation and prediction algorithms and techniques based on historical data range from simple models, based on average or exponential smoothing, to complex ones, such as autoregressive, moving average, integrated, and combinations thereof [1]. The load for MMORPGs, however, is determined not only by the number of players, but also by what they do in the virtual world, since different activity patterns generate widely different load [2].

A model for network load and server utilization of MMOG systems has been proposed Yee and Cheng in [3]. They describe the relationship between the number of active players and input and output traffic, as well as the processor utilization and claim that the dependency is strongly linear. A later work by Lee and Cheng in [4] has been applied for energy and hardware resource usage optimization, based on past player behavior. The authors focus on location of players in the virtual world and develop a zone-based server consolidation strategy.

An notable approach for load calculation for MMORPGs has been proposed by Nae and Prodan [5], [6]. They explore a neural networks based load prediction for MMORPGs, and define formulae for calculation of required processing time based on different states of the virtual world.

In our previous work, we have shown that MMORPG player activity patterns can be categorized, and that each category may be represented by a corresponding mathematical model. The proposed action categories have been termed Trading, Raiding, Player vs Player (PvP) combat, Dungeons, and Raiding [7]. In this work we extend the concept of *interaction complexity* applied for load prediction by Nae and Prodan by taking into account MMORPG player behavior categories, and asses the prediction accuracy of the proposed approach.

TABLE I
Interaction complexity per action category

Behavior category	Players vs Players	Players vs NPC	Number of NPCs	Network load [kbit/s]
Questing	$O(n)$	$O(n * log(n))$	$n \leq 5$	11.4
Trading	$O(n)$	$O(n)$	$N \leq 20$	8.1
Dungeons	$O(n^2)$	$O(n^2)$	$N \leq 20$	18.3
PvP combat	$O(n^3)$	$O(n)$	$n = 0$	24.1
Raiding	$O(n^2 * log(n))$	$O(n^3)$	$N \leq 40$	32.0

II. Interaction complexity

Our approach is based on mapping of action categories to interaction complexities in load calculation formula proposed by [5]. The mapping is presented in Table I and it is based on the specific characteristics of the action categories. For example, PvP combat is characterized by only involving human players (no NPCs) and the most complex interaction between players, while Raiding has complex interactions between players and NPCs as well as amongst players themselves.

III. Prediction model

Prediction of the load is based on complete *history dataset* H of player behavior on a single shard. The time length of the *pattern* we want to capture is P (e.g., an hour, a day, a month). P is divided into discrete intervals i (e.g., seconds, minutes, hours) at which we observe the number of active players in each action category.

Prediction of the number of players for each action category is based on an algorithm which adapts the parameters calculated from the entire history dataset H with the data from the last *window* L which has the same size as the pattern P.

For each discrete interval i we calculate:

$$R(i) = Max\{H(i), L(i)\} \cdot L(i)/H(i) \cdot Max(L) \quad (1)$$

where $R(i)$ is the predicted number of players, $H(i)$ is the historical information, $L(i)$ is the information from the last window, and $Max(L)$ is the maximal number of players in the last window.

978-1-4673-1567-8/12/$31.00 ©2012 IEEE

This approach makes the algorithm prone to overestimation, but results in quick adaptation based on a sliding window. Also, the algorithm is simple enough to perform the calculation in real time. For example, we can recalculate the prediction with new data in the order of minutes, and thus enable quick adaptation in case of "flash mobs" in the game.

Based on the results of the prediction of the number of players for each action category, we calculate two types of load: a) processing load, and b) network load. We use formula for CPU load of a distributed game server (comprising several machines) as defined in [6].

$$t_C = (N + BE + p_{ui} \cdot p_{ci} \cdot f(IC, IC) + p_{ui} \cdot p_{ei} \cdot f(IC, BE)) \cdot t_u \quad (2)$$

where t_C is the overall time spent in calculating one tick of the virtual world; N is the overall number of users present; $f(a, b)$ is the interaction function, where parameters a and b represent the numbers of interacting entities (avatars, NPCs); IC is the number of avatars (i.e, human users) interacting with other entities; BE is the overall number of NPCs; p_{ui} is the ratio between the time necessary for one entity update and the time for computing one interaction (in percentage); p_{ci} is the percentage of users that are performing a certain action in that moment; p_{ei} is the percentage of NPCs that are performing a certain action in that moment; and t_u is the update time of entity states received from (sent to) another machine.

To take into account the level of interaction in given action category, the interaction function f applies the interaction complexities as defined in Table I.

IV. Experiments

We perform two experiments, prediction for a "typical" scenario in which the simulation log is based on the normal simulation parameters. Also, we preform an experiment in which we observe a scenario which simulates a surge in the number of users, and also a different behavior pattern.

For both experiments, we use the previously developed user behavior simulator [2] as a source of the history on which the prediction is based. Log files obtained from the simulation are used as an input for the predictor. By changing the parameters of the behavior simulation we can create logs of different scenarios. This enables various tests besides usual behavior, e.g., addition of a patch with a new raiding instance significantly increases the popularity of Raiding category, flash mobs of certain action categories, etc. Eight days are simulated and the results are stored in the the log. Seven days are used as a history data to train the predictor, and the eight day is used for verification. The load of the next day is predicted with interval size of one minute.

Figure 1 shows the an example of CPU prediction results without (top) and with (bottom) behavioral information. Three lines are depicted, the predicted values, the values of the actual load (calculated from the eight day), and the reserved values (15% additional resources over the predicted).

Results indicate that there is a very significant difference between prediction which takes only a number of users into account (i.e, one interaction complexity), and the prediction which takes into account user behavior (i.e., multiple interaction complexities). The discrepancies in this scenario are reaching even 30%. This suggests, that for correct estimates of load player behavior needs to be taken into account.

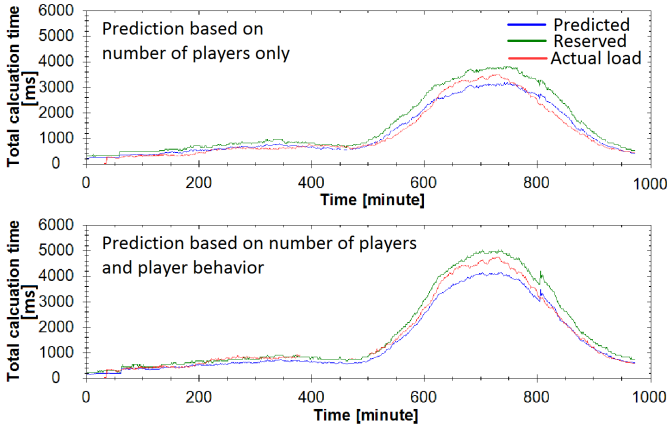

Fig. 1. Prediction results with (top) and without (bottom) user behavior

The load prediction showed to be quickly adaptive to the changes in the last period yet capturing the patterns. Even with the spikes with 200% of increase in the number of players, the prediction would adapt the numbers for the next discrete interval. The downside is that the algorithm would predict high load for the whole next period, which introduces a significant error if the spike in the number of users was just temporary.

V. Conclusion

In this paper we confirmed that application level behavior has significant impacts on the both CPU and network load of a MMORPG system through a development of a tool for load prediction of MMORPGs. For our future work we aim to test more complex algorithms, as well as to implement a real time prediction which could be recalculated every discrete interval.

References

[1] G. Box, G. M. Jenkins, and G. Reinsel, *Time Series Analysis: Forecasting and Controls*. New York: Prentice-Hall, 1994.

[2] M. Suznjevic, I. Stupar, and M. Matijasevic, "A model and software architecture for MMORPG traffic generation based on player behavior," *Multimedia Systems*, 2012. [Online]. Available: http://www.springerlink.com/openurl.asp?genre=article&id=doi:10.1007/s00530-012-0269-x

[3] M. Yee and L. Cheng, "System-performance modeling for massively multiplayer online role-playing games," *IBM Systems Journal*, vol. 45, no. 1, pp. 45–58, 2006.

[4] Y.-T. Lee and K.-T. Chen, "Is server consolidation beneficial to MMORPG? A case study of World of Warcraft," in *IEEE International Conference on Cloud Computing*, 2010, pp. 435–442.

[5] R. Prodan and V. Naea, "Prediction-based real-time resource provisioning for massively multiplayer online games," *Future Generation Computer Systems*, vol. 25, pp. 785–793, 2009.

[6] V. Nae, A. Iosup, and R. Prodan, "Dynamic resource provisioning in massively multiplayer online games," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, pp. 380–395, 2011.

[7] M. Suznjevic, O. Dobrijevic, and M. Matijasevic, "MMORPG player actions: Network performance, session patterns and latency requirements analysis," *Multimedia Tools and Applications*, vol. 45, no. 1-3, pp. 191–241, 2009.