Modeling System Performance in MMORPG

Gao Huang, Meng Ye, Long Cheng
IBM China Research Lab
Shangdi, Beijing 100085, China
EMAIL: huanggao, yemeng, lcheng @cn.ibm.com

***Abstract*—Massive multiplayer online games are becoming popular and prosperous rapidly. Although many papers have been published on system performance for online games in recent years, most of them are focusing on network perspective in First Personal Shooter (FPS) or strategy games. This paper describes a method to modeling network traffic and game server performance in a standard MMORPG (Massive Multiplayer Online Role Play Game). Experimental results show that there exists strong linear relationship between performance metrics in server side and the concurrent player number online. Utilization of two kinds of resources, including network traffic and server, can be calculated via using the concurrent player number as a determinative variable. Player-related distribution patterns are also analyzed which form the basis for resource provisioning and load balancing.**

Keywords- Online Game, MMORPG; System Performance; Concurrent Player Number

I. INTRODUCTION

Online games have become increasingly popular in recent years. According to the report [1] of CCPA and IDC, Chinese online game players increased to over 20 millions, with a revenue of 1.32 billion RMB(about 0.15 billion USA dollars) in 2003. Unlike the producing-selling business model of standalone games, MMOG (Massive Multiplayer Online Game) is a service business in which game-playing experience would be a critical factor for success. Since most MMOG available in market are based on some sorts of Client-Server architecture, the performance of MMOG game servers will significantly influence game-playing experience. Modeling system performance in MMOG, however, becomes an essential work for promoting customer satisfaction level. Many papers have been published on this topic, yet until now, most of the papers focused only on network traffic in FPS (Fist Person Shooter) games or strategy games [16, 18, 22, 23, 24], and few analysis has been done to the most popular online game style in the Asian market –Massive Multiplayer Online Role Playing games(MMORPG) [2]. FPS and strategy games are different from MMORPG in many aspects. The most obvious difference is that MMORPG has much more simultaneous players in same game world than have other kinds of games. In this paper, system performance of a standard MMORPG game, which is one of the most popular titles in China, is analyzed.

A highly simplified view of MMORPG architecture consists of four parts [2]: client side, Internet connection, game server side and account management (Fig.1). The use of web interfaces for account management is already quite standard, which is a typical web application and is not related to game running time. From the perspective of game world performance, a game service provider would care more about game server side, since performance in client side is similar to that of standalone PC games. Although new problems arise in client side of online games, i.e., dead reckoning [3, 4], they are not special issues for MMORPG games, but for all online games. Internet performance is another important topic for MMORPG performance. We will give a brief discussion of it in the next section, but won't go into details in that Internet is statistically multiplexed by various applications.

Currently, the standard MMORPG infra-structure is a typical multi-tier Client/Server architecture (Fig.1). A proxy server farm communicates with all players. Usually a load balancing algorithm, such as round robin, is used to select a proxy for a player who wants to join the game. Since a game world in MMORPG is too large for a single server to support all the activities in it, it is normally divided into several small ones, and a cluster of game servers serve the participated game world together. Proxy server acts as a dispatcher between clients and game servers, according to which server contains the client. A processing cycle usually starts with a client sending a message to proxy, and the proxy in turn dispatches the command to the corresponding game server. Game server performs the game logic, modifies the status of game world and sends back the updated player's status to all stakeholders based on AOI (Area of Interest) management [5]. Besides of the back-and-forth messages caused by player actions, some system messages, such as announcement of game master and handshaking between client and server, are also delivered via the same channel.

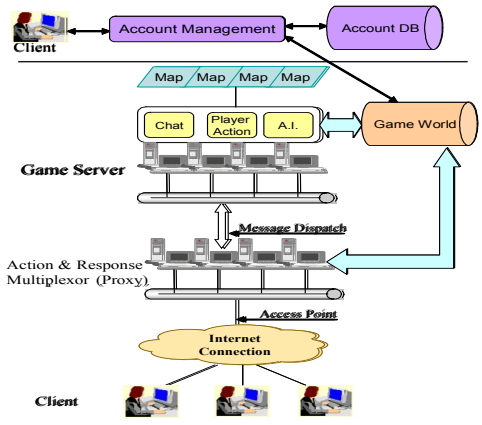


Figure 1. Standard MMORPG infra-structure

0-7803-8798-8/04/$20.00 ©2004 IEEE

II. RELATED WORK

System performance is a broad topic which covers almost all the aspects of online game, from application architecture design such as P2P and C/S, to QoS control [6, 7]. Although various architectures have been discussed in some research papers [8, 9, 10], the most popular one in market is still the Client/Server architecture. Under C/S architecture, system is composed of four parts as described in section I. Although the Internet performance has been improved a lot in the past years, it still might be a bottleneck for MMOG players. IntServ [12] was proposed to provide per-flow QoS control; DiffServ [13] was proposed for specific kinds of applications based on PHB definition; Chen, Gavros, et al. provide overviews on those two topics [14, 15]. The problem with these solutions is that they still can not be widely used in current Internet backbone.

Being a sub-area of system performance, issues related to on-line game network traffic have been studied for some time. Borella and Farber set up a model of game network traffic [16, 17]. Chang et al. give us an understanding of the player's behavior related to network traffic [18, 19, 20]. Some other topics, such as delay and bandwidth have also been discussed [21, 22, 23, 24]. All those studies gave us a further understanding of online game network traffic, however, most of them were focusing on FPS or strategy games, and little was done on MMORPG games.

The rest of the paper is structured as follows. In Section III we would have a brief introduction on problems met in MMORPG games at first; then the model of system performance is discussed by introducing concurrent player number as a determinative variable, which consists of two parts: network traffic model and server performance model; finally, distribution patterns of player related metrics are discussed. Section IV analyzes the experiments results. Last section is a summary and suggestions for future work.

II. SYSTEM PERFORMANCE ANALYSIS

A. Performance Issues in MMORPG

In MMORPG, a game service provider usually runs many game worlds which in essential are isolated instances of the same game. Players select a game world and play in it. The number of concurrent players is a popular metric for the measurement of game server performance. Normally, as the number grows up, the performance of game worlds is degraded. The MMORPG game we analyzed can support about 2000 to 3000 maximum concurrent players, which uses four game servers and four proxy servers (Fig.1). If the concurrent player number exceeds this point (Fig.2), some problems arise:

1) Long response time: After sending a command to server, the player has to wait for a long time before receiving the response. For cheating-proof purpose, all game logics are processed in server side. Although some techniques, such as dead reckoning, could be used, they only provide partial solutions. Besides poor user experience, some other problems, such as fairness between players, would be caused [19, 23, 25].

2) Action roll back: If a client cannot receive the response from servers within a negotiated interval, action roll back occurs, and therefore, one would see that s/he moves forward and backward in screen.

3) Server related issues: Servers would have higher malfunction probability under heavy load, such as writing errors to database, which causes the lost of player's status if one logouts at that time.

Two techniques, load balancing and resource provisioning, are widely used to solve or soften the above performance issues. No matter which technique is used, a basic problem is how to model system performance in server side.

B. Network Traffic in MMORPG

Game network has two kinds of traffic, i.e., game logic related traffic and ancillary function related traffic. All these kinds of traffic are comprised of two parts: incoming traffic and outgoing traffic.

Network traffic is mainly composed of game logic related network traffic in MMORPG. When a player's status is changed, not only this player should be notified, but all the players who could see him/her in their view scope should be notified [5]. So the network traffic in one processing cycle is composed of one command sent from the player and some outgoing messages sent to related players. Therefore, the outgoing traffic is not only related to the concurrent player number, but also related to the geometry distribution of all the players in the game world. Fortunately, in MMORPG, the other players included in one player's view scope are relatively small in most of the time, so the network traffic in one processing cycle is relatively stable. If every player's behavior is independent with others (In most cases it is true in MMORPG), the commands sent by all the players in one time slot is only decided by the concurrent player number, which means the game logic related network traffic can be described by (1), where $\lambda(t)$ is concurrent player number at time t.

$$N(t) = n_{\lambda} \times \lambda(t) \tag{1}$$

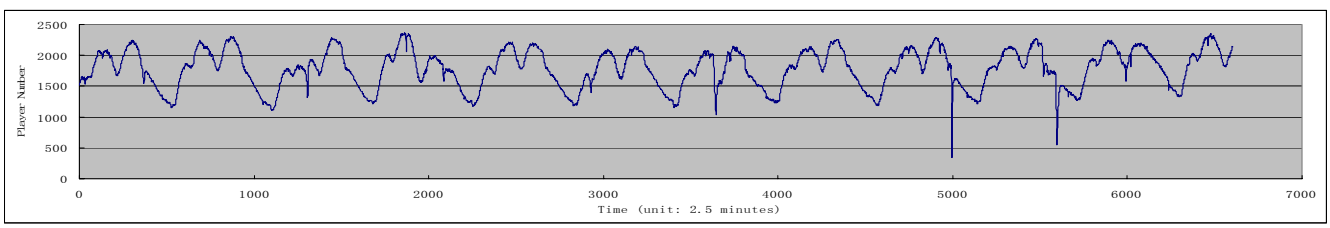


Figure 1. Example of concurrent player number variation which is collected in 11 days.
x axis is time whose sample rate is 15 minutes, y axis is the concurrent player number which vary between 1000 and 2500.

Some network traffic is caused by ancillary functions, two typical ancillary functions in MMORPG, connection maintenance and chat, are used here to analyze this kind of game network traffic. If the game server has not received any message from a client for a specific interval, a request is sent to the client to query its status such as whether the client is still online. Therefore the traffic caused by one maintenance message is fixed, thus the connection maintenance traffic is in proportion to the concurrent player number ((1) with a different n_λ value). Chat function, which is the process that a player broadcast messages to all the other players, is supported in most of current MMORPG games. It is obviously that the traffic caused by one broadcast process is proportional to concurrent player number: one incoming message and $\lambda(t)$ -1 outgoing messages. So the whole traffic caused by all the chat messages is proportional to the square of concurrent player number (equation 2).

$$N(t) = \varepsilon \times \lambda(t)^2 \tag{2}$$

By putting all the factors together, we have:

$$N(t) = n_\lambda \times \lambda(t) + \varepsilon_\lambda \times \lambda(t)^2 \tag{3}$$

Since the traffic caused by chat messages is relatively small, the model can be simplified to (1).

W. Feng et al. show in [29] that the game network traffic is highly predictable, with a Hurst parameter around 0.5. Therefore, no long-range dependency exists. So once the concurrent player number is obtained, the network traffic can be predicted by using (1) or (3).

C. Server performance in MMORPG

It is well known that in traditional web applications, server performance can be modeled by the arrival rate [32, 33]. For most of the applications, this model can be expressed in a linear function:

$$U(t) = \lambda(t) * u_\lambda + b \tag{4}$$

Where λ is the arrival rate, usually it is defined as the request number from clients. b indicates the server resource used by the functions deployed in server side that are not related to any requests. u_λ is the resource utilization rate used by one request. Vasilesco gives the details of how to calculate these metrics [33].

However, in MMORPG, server logic is very similar to that of traditional web applications. A command is send to server, then it is processed by servers and results are sent back.

A factor that would undermine the linear relationship between server performance and concurrent player number is the load balancing algorithm. The concurrent player number is defined as the number in a whole game world, and these players are distributed to each proxy servers and game servers by the load balancer. If we tokenize the player number on each game server as $\lambda_g^i(t)$, the player number on each proxy server as $\lambda_p^i(t)$, we have:

$$\lambda(t) = \sum_{i=1}^{4} \lambda_g^i(t) = \sum_{i=1}^{4} \lambda_p^i(t) \tag{5}$$

Therefore, a point that has not been considered is: How does the load balancer distribute the players to each server? If the load balancer distributes the players to each server randomly, it can not be guaranteed that the data will exhibit a linear relationship even if each of them follows (6)

$$\lambda^i(t) = U(t)_i / u_{\lambda i} - b_i / u_{\lambda i} \tag{6}$$

Here we define a basic characteristic of the load balancer as proportion-consistent. A load balancer is proportion-consistent if the algorithm used by it dispatches the traffic to each resource proportionally, and the proportion does not change over the time. Then we would have the following conclusion:

Conclusion 1: If each server's performance has a linear relationship to the player number on this server, and the load balancer is proportion-consistent, then every server's performance also has a linear relationship with the player number in the whole game world. Since the proof is straightforward, it is omitted it here.

The load balancer in our system uses a WRR (Weighted Round Robin) algorithm to dispatch the players, which is obviously proportion-consistent.

D. Player Distribution Analysis

Fig. 2 shows an example of the concurrent player number in about 11 days. The data in one day is picket up and shown in Fig.3, which indicates that there is a clear pattern in the concurrent player number variation. Three peak time points in one day are around 11:00am, 4:00pm and 9:30pm. The values of last two peak points are obviously higher than that of the first peak point. Some tips might be got from the figure on how to do resource provisioning on server side. Since network traffic and server performance both can be decided by concurrent player number, the network and server resource should be provisioned to this game world at the three peak load points and de-provisioned when the load comes down.

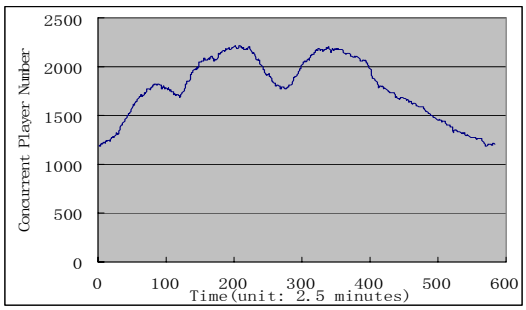


Figure 2. Concurrent player number variation in one day, the start point is 7:30am

By collecting the data in about 100 days (Fig.2), the distribution of the concurrent player number is analyzed (Fig.8 (a) and (b)). A surprising point is found that the concurrent player numbers varying from 1 to 50 have very high incidents (sample data in Fig.8 (a)). However, it indicates that the game world is in a warming up procedure. Right after a game world

is launched, few players would be aware of it until some time later, and therefore, the concurrent player numbers in these time slots are small values. After removing these starting points, it is found that the distribution can be depicted perfectly (bold line in Fig.8 (a)) by a Weibull distribution (equation (7) and (8)).

$$P(x) = \alpha\beta^{-\alpha}x^{\alpha-1}e^{-(x/\beta)^{\alpha}} \quad (7)$$

$$D(x) = 1 - e^{-(x/\beta)^{\alpha}} \quad (8)$$

III. EXPERIMENTTAL RESULT

A standard MMORPG game, which is one of the most popular titles in China, is analyzed by using the model described in the last section. Following is the standard configuration of a game word of this MMORPG. Each game world would have a bandwidth of 32Mb to Internet.

TABLE I. CONFIGURATION OF THE SERVER SIDE OF THE MMORPG GAME

Server Type	Proxy Servers	Game Servers
CPU	1.8G Hz	1.8G Hz
Memory	2GB	2GB
Hard Disk	30GB	60GB
Network interface	100M(Internet)/ 1000MB(LAN)	1000MB

A. *Network traffic in MMORPG*

In order to analyze the network performance, a heavy-load game world is selected for monitoring, which has a concurrent player number vary from 1500 to 2500. By attaching it to the concurrent player number calculated from the log information in database, their interrelationship can be found. First, linear model is evaluated by calculating the correlation coefficient Cov between concurrent player number and network traffic according to (9).

$$Cov(\lambda(t), N(t)) = \frac{\sum(\lambda(t) - E(\lambda(t)) * (N(t) - E(N(t)))}{\sqrt{D(\lambda(t))D(N(t))}} \quad (9)$$

Secondly, a robust regression algorithm is used to get the parameter n_λ in (1), then the two curves $N(t)$ and $\lambda(t) * n_\lambda$ can be plotted together and compared. A detailed discussion of the algorithm can be found in [30, 31].

Two network traffic metrics were analyzed: bytes/second and packets/second. Table II shows the results, which shows a strong linear relationship between them. Fig.4 shows the robust regression results. Although network traffic is highly fluctuated, its relationship with concurrent player number strongly follows (1).

TABLE II. LINEAR RELATIONSHIP AND PARAMETERS OF NETWORK TRAFFIC AND CONCURRENT PLAYER NUMBER, Cov IS THE CORRELATION COEFFICIENT CALCULATED ACCORDING TO (9), n_λ IS THE PARAMETER IN (1) ESTIMATED BY THE ROBUST REGRESSION ALGORITHM.

Metrics Name	Cov	n_λ	$1/n_\lambda$
Bytes	0.8834	714.29	0.0014
Packets	0.8459	7.57	0.1321

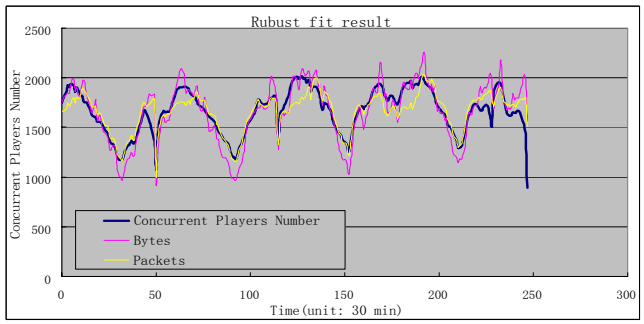
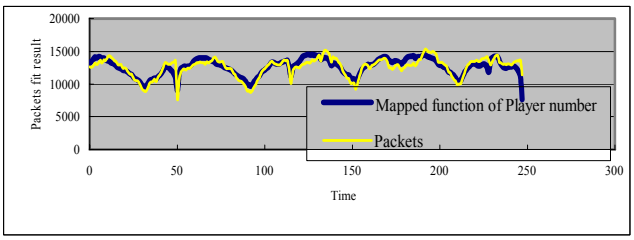


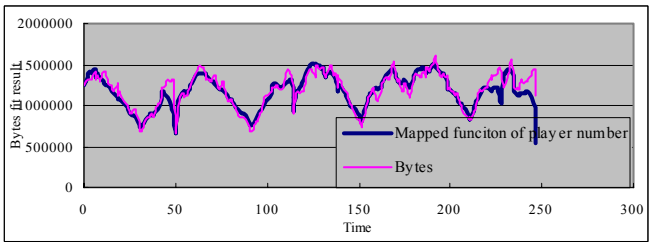
Figure 3. Robust regression results between network traffic and concurrent player number, X axis is time with a sample rate of 30 minutes. The bold (blue) line is the concurrent player number, light colored (yellow) line is mapped from the packets traffic to concurrent player number as $\lambda_p(t) * 0.1321$ (see table 2) and $\lambda_p(t)$ is the packet traffic; similar to this, middle colored (red) line is mapped from the bytes traffic to concurrent player number as $\lambda_b(t) * 0.0014$ and $\lambda_b(t)$ is the bytes traffic

Finally, in order to see whether the proportional model is good enough, the square model is also evaluated by calculating $Cov' = Cov(N(t)', N(t))$, where $N(t)'$ is tokenized as $n_\lambda \times \lambda(t) + \varepsilon_\lambda \times \lambda(t)^2$. The results are shown in Table III, it can be seen from the column $\frac{Cov' - Cov}{Cov}$ that this model doesn't surpass the proportional model. Fig.5 plots the robust fit results of this model, it can be seen from the figure that the accuracy of two models are almost the same.

An interesting point in Table III is that the coefficient ε_λ of packets is negative, which means that the square factor of concurrent player number has a negative effect on packets traffic. In fact, in MMORPG, merging packets is a widely used technique to increase the network efficiency. This technique decreased a great deal of packets and offset the square factor of concurrent player number.



(a) Player Number/Packet fit result of the model includes the square factor



(b)Player Number/Bytes fit result of the model includes the square factor

Figure 4. Fit results of the model includes the square factor

TABLE III. THE MODEL OF NETWORK TRAFFIC AND CONCURRENT PLAYER NUMBER WHICH INCLUDES THE SQUARE FACTOR, Cov' IS THE CORRELATION COEFFICIENT CALCULATED AS $Cov' = Cov(N(t)', N(t))$, n_λ AND ε_λ ARE THE PARAMETERS IN (3) ESTIMATED BY THE ROBUST REGRESSION ALGORITHM.

Metrics Name	Cov'	$\frac{Cov'-Cov}{Cov}$	n_λ	ε_λ
Bytes	0.8832	-0.0002	467.00	0.1400
Packets	0.8538	0.0086	9.64	-0.0012

B. *Server Performance in MMORPG*

In order to evaluate server performance model, windows performance monitor is used to record the CPU utilization of the related game processes in every 5 seconds, then they were summed together to get the whole utilization on each server (Proxy and Game Server). Fig.6 is an illustration with a sample rate of 5 seconds, which shows that the game daemons on servers exhibit extreme turbulence, the reason is that the game logic and programming style have a great impact on small sample rates.

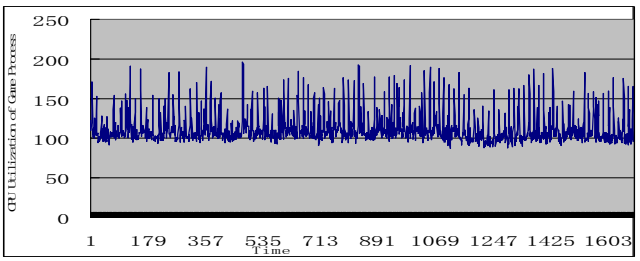
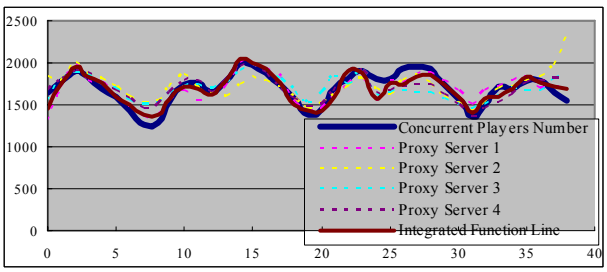
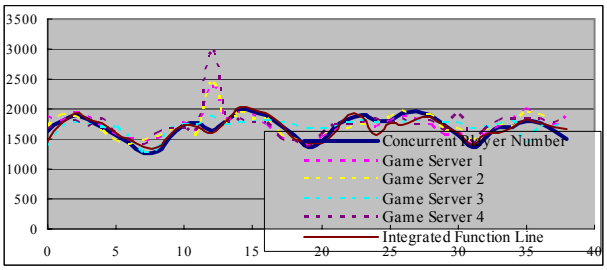


Figure 5. CPU utilization of Game Process, X axis is the time dimension with a sample rate of 5 seconds. Y axis is the sum of CPU Utilization used by all game related processes on one server



(a) plots the curves of four game servers



(b) plots the curves of four proxy servers

Figure 6. Linear relationship between server performance and concurrent player number,. The bold (blue) line is the concurrent player number, the thin (umber) line is the integrated function calculated by (11), while the other four dotted lines are the curve of (10) for the servers

Two metrics, server performance and player number, are attached together with a sample rate of one hour to eliminate disturbance, and the server performance is averaged during the interval. The procedure is similar to that of network traffic analysis: first, correlation value $Cov(\lambda(t), U(t))$ between CPU utilization and concurrent player number is calculated; Secondly, the robust regression algorithm is used to find the parameter of (10), which is a transformation of (4). (For we want to plot the lines in one figure):

$$\lambda(t) = U(t)/u_\lambda - b/u_\lambda \tag{10}$$

Where $\lambda(t)$ is the concurrent player number; $U(t)$ is the server performance; u_λ is the server's CPU utilization cost by one player; b dedicates to work of daemons on server.

Table IV summarizes the results which show that almost all of the servers have a strong linear relationship with concurrent player number except game server 4. Fig.7 plots the robust regression result of the servers. From it, it can be seen that although the correlation coefficient of game server 4 is 0.3491, its server performance still fits (10) very well. If some outliers points, less than 5% of the original data set, are removed, game server 4 also exhibits a high linear relationship with the concurrent player number with a correlation value of 0.6731.

TABLE IV. LINEAR RELATIONSHIP BETWEEN THE SERVER PERFORMANCE AND THE CONCURRENT PLAYER NUMBER, THE PARAMETERS OF (10) ARE ESTIMATED BY ROBUST REGRESSION ALGORITHM.

Server ID	Cov	$1/u_\lambda$	$-b/u_\lambda$
Proxy Server1	0.8233	38.65	819.18
Proxy Server2	0.503	27.9	1063.7
Proxy Server3	0.6856	20.7	1189.4
Proxy Server4	0.8019	31.79	976.84
Game Server1	0.648	39.4	-2561.1
Game Server2	0.6258	45.7	-2827.2
Game Server3	0.6049	9.388	698.44
Game Server4	0.3491	76.7	-6599

Since each game server is bound to one part of the whole map, the players on one server is equal to the players on the partial map of the server. Therefore whether the load balancing algorithm for game server is proportion-consistent is decided by the geometry distribution of players on the whole map. It is why the correlation coefficients of game servers are poorer than those of proxy servers (Table IV). On the other hand, if every sever follows (6), the concurrent player number of the whole game world would follow (11). Table V shows the results, $Cov(\lambda', \lambda)$ indicates that the linear relationship is improved (umber line in Fig.7).

$$\lambda(t)' = \sum_{i=1}^{4} U(t)_i / u_{\lambda i} - \sum_{i=1}^{4} b_i / u_{\lambda i} \tag{11}$$

0-7803-8798-8/04/$20.00 ©2004 IEEE

TABLE V. LINEAR RELATIONSHIP BETWEEN INTEGRATED RESULTS (11) AND CONCURRENT PLAYER NUMBER, THE PARAMETERS ARE ESTIMATED BY ROBUST REGRESSION ALGORITHM

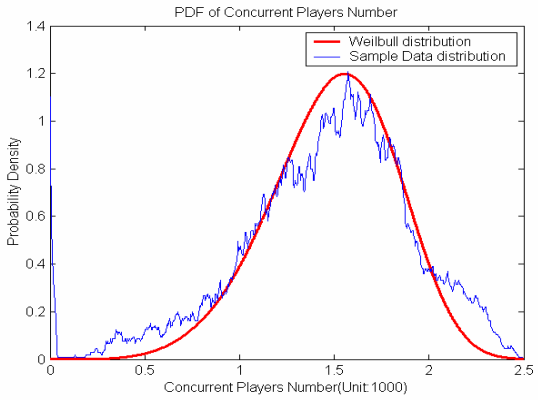
Server Group	$Cov(\lambda,\lambda')$	$1/u_{\lambda 1}$	$1/u_{\lambda 2}$	$1/u_{\lambda 3}$	$1/u_{\lambda 4}$	$-\sum_{i=1}^{4} b_i / u_{\lambda i}$
Game Servers	0.81	32.8	10.4	6.9	19	-1554.9
Proxy Servers	0.90	24.2	-5.4	-6.8	28.9	785.9

C. *Player Distribution Analysis*

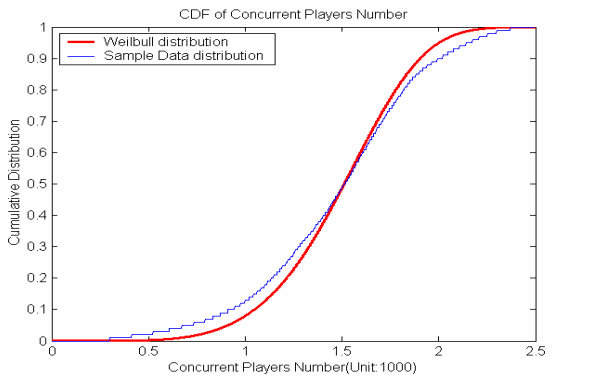
MLE (Max Likelihood Estimation) algorithm is used to estimate the parameters of Weibull distribution (Table VI). Fig.8 (b) shows the CDF of concurrent player number (the starting points are removed). The maximum concurrent player number is 2451. PDF and CDF of the model are shown in (7) and (8).

TABLE VI. CHARACTERISTICS OF PLAYER-RELATED METRICS, α AND β ARE THE PARAMETERS OF WEIBULL DISTRIBUTION, MAXIMUM VALUE OF CORRESPONDING METRICS IS GET FROM THE DATA IN ABOUT 100 DAYS

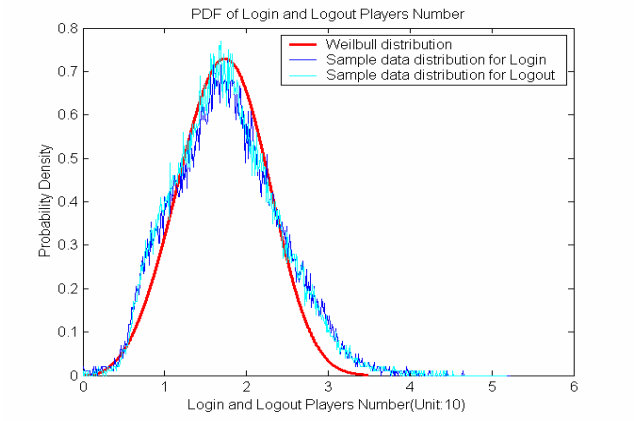
Metrics Name	α	β	Maximum value
Concurrent Players	5.1828	0.081	2451
Login Players	3.6212	0.097	44
Logout Players	3.6353	0.096	57



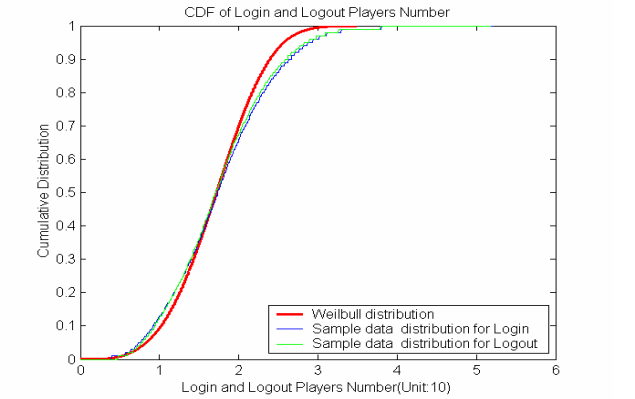
(a) PDF of Concurrent Players Number, bold (red) line is the PDF of Weibull distribution, thin (blue) line is the PDF of the sample data for concurrent player number



(b) CDF of Concurrent Players Number, bold (red) line is the CDF of Weibull distribution, thin (blue) line is the CDF of the sample data for concurrent player number



(c) PDF of login and logout Players Number,bold (red) line is the PDF of Weibull distribution, saturated thin (blue) line is the PDF of sample data for login player number, light colored (green) line is the PDF of sample data for logout player number



(d) CDF of login and logout Players Number, bold (red) line is the CDF of Weibull distribution, saturated color thin (blue) line is the CDF of sample data for login player number, light colored line (green) is the CDF of sample data for logout player number

Figure 7. Concurrent Players Number/Login Players/Logout Players of Weibull distribution

Fig.8 (c) and (d) are the distributions of login player number and logout player number in every second. Weibull distribution is also used to describe this two metrics' distribution. Though from the curve, it can be argued that a Gaussian distribution is good enough, it is not adopted here, since the login and logout number vary drastically over the time and do not have a centric mean. The maximum number for login players and logout players in one second is 44 and 57. The variance of (Login number – Logout number) causes the jitter in Fig.8.

Another interesting point of the distribution is that the heavily tailed phenomenon is not so evident (Hurst Parameter is 0.563), as it is in many network applications [34].

IV. CONCLUSION AND FUTURE WORK

This study has proposed a performance model in MMORPG. With the data collected from real MMORPG

operation, it is demonstrated that the performance metrics in server side has a strong linear relationship with concurrent player number. By using this simple model, the utilization of two kinds of resources, including network bandwidth and server, can be calculated. In this paper, some important player-related distribution patterns are analyzed, the conclusions can be used to form the basis for improving the system performance. Though the analysis is based on a specific MMORPG, its results can also be applied for other MMORPG, because most of the target games in market are built with the same architecture.

The scope of this study, however, is limited, and much of the current conclusions are based on pure statistics. We will continue the investigation in the following aspects:

- User behavior analysis in MMORPG and the geometry distribution pattern in the game world. This study will help us answer questions met in MMORPG, such as how to divide the whole map into small pieces.
- More performance analysis of different MMORPG and other kinds of online games.
- How to use the results got in this study to improve system performance? Such as using the results in dynamic load balancing and resource provisioning.

REFERENCES

[1] CGPA, IDC,"Chinese game industry report of 2003"

[2] Jarett A, Estanislao J, Dunin E, MacLean J, Robbins B, Rohrl D, Welch J, Valadares J "(2003) IGDA Online Games White Paper",http://www.igda.org/online/IGDA_Online_Games_Whitepaper_2003.pdf

[3] S. K. Singhal, "Effective Remote Modeling in Large Scale Distributed Simulation and Visualization Environments," Ph.D Dissertation, Department of computerscience, Stanford University, 1996.

[4] Y. Bernier. "Latency Compensating Methods in Client/Server In-game Protocol Design and Optimization", Proceedings of the Game Developers Conference, February 2001

[5] Jouni Smed, Timo Kaukoranta, Harri Hakonen,"A Review on Networking and Multiplayer Computer Game" Turku Centre for Computer Science,2002

[6] Bauer, D., Rooney, S., Scotton, P.: Network Infrastructure for Massively Distributed Games. In: NetGames 2002 -- First Workshop on Network and System Support for Games, Braunschweig, Germany (2002).

[7] Jouni Smed, Timo Kaukoranta, Harri Hakonen,"Aspects of Networking in Multiplayer Computer Games",Proceedings of International Conference on Application and Development of Computer Games in the 21st Century.

[8] C. Diot and L. Gautier, "A distributed architecture for multiplayer interactive applications on the internet," IEEE Network Magazine, vol. 13, no. 4, July/August 1999.

[9] Laurent Gautier and Christophe Diot. Mimaze, "a multiuser game on the internet". Technical Report RR-3248, Institut National de Recherche en Informatique et en Automatique (INRIA), September 1997

[10] R. A. Bangun and H. P. Beadle. "A network architecture for multiuser networked games on demand" In Proceedings of the 1997 International Conference on Information, Communications and Signal Processing, pages 1815-1819, Singapore, September 1997.

[11] Lothar Pantel, Lars C. Wolf: "On the suitability of dead reckoning schemes for games". NETGAMES 2002

[12] Shenker, S., Partridge, C and Guerin, R., "Specification of Guaranteed Quality of Service", IETF RFC 2212, 1997

[13] Blake, R., Black, D., Carlson, M., Davies, E., Wang, Z. and Weiss, W., "An Architecture for Differentiated Services",IETF RFC 2475, December 1998

[14] Shigang Chen and Nahrstedt, K., "An overview of quality of service routing for next-generation high-speed networks: problems and solutions". IEEE Network,November/December 1998.

[15] Panos Gevros, Jon Crowcroft, Peter Kirstein, and Saleem Bhatti, "Congestion Control Mechanisms and the Best Effort Service Model". IEEE Network,May/June 2001.

[16] M. S. Borella. Source Models of Network Game Traffic. Computer Communications, 23(4):403--410, Feb 2000.

[17] Johannes Farber, "Network Game Traffic Modelling", Proceedings of the first workshop on Network and system support for games, 2002.

[18] Francis Chang, Wu-chang Feng, "Modeling Player Session Times of On-line Games",ACM NetGames 2003,May,2003

[19] T. Henderson, "Latency and User Behaviour on a Multiplayer Game Server," Proceedings of the 3rd International Workshop on Networked Group Communication (NGC), 2001

[20] R. A. Bangun, E. Dutkiewicz, and G. J. Anido. "An analysis of multi-player network games traffic".In Proceedings of the 1999 International Workshop on Multimedia Signal Processing, pages 3-8, Copenhagen, Denmark, September 1999

[21] Pellegrino, Dovrolis, "Bandwidth requirement and state consistency in three multiplayer game architectures," NetGame 2003

[22] N. Sheldon, E. Girard, S. Borg, M. Claypool, E. Agu. "The Effect of Latency on User Performance in Warcraft III", Technical Report WPICS-TR-03-07, Computer Science Department, Worcester PolytechnicInstitute, March 2003

[23] G.Armitage. "An Experimental Estimation of Latency Sensitivity in Multiplayer Quake 3", 11th IEEE International Conference on Networks (ICON 2033), Sydney, Australia, September 2003

[24] Lothar Pantel, Lars C. "Wolf On the impact of delay on real-time multiplayer games" , NOSSDAV, 2002

[25] Sung-Jin Kim, Falko Kuester, K. H. Kim: "A Global Timestamp-Based Scalable Framework for Multi-Player Online Games". ISMSE 2002

[26] J. C. R. Bennett, H. Zhang, "Hierarchical Packet Fair Queueing Algorithms", IEEE/ACM Transactions on networking, Vol. 5., No. 5, October 1997

[27] PAREKH, A. K. and GALLAGER, R. G. (1993). "A generalized processor sharing approach to flow control in integrated services networks: The single-node case". IEEE/ACM Transactions on Networking 1 344-357.

[28] PAREKH, A. K. and GALLAGER, R. G. (1994). "A generalized processor sharing approach to flow control in integrated services networks: The multiple node case". IEEE/ACM Transactions on Networking 2 137-150.

[29] W. Feng, F. Chang, J. Walpole, "Provisioning On-line Games: A Traffic Analysis of a Busy Counter-Strike Server", In Proceedings of the Internet Measurement Workshop, November 2002

[30] P. J. Rousseeuw, and A.M. Leroy, "Robust regression and outlier detection". John Wiley & Sons, 1987

[31] Barnett, V. and Toby, L., (1994), "Outliers in statistical data", 3rd ed., Wiley

[32] Alex Vasilesco,"Forecasting CPU Utilization: a Simple Saturation Model in a UNIX Web Applications Server Environment"

[33] Menascé, D. A., Almeida, "Capacity Planning for Web Performance. Metrics, Models, &Methods" Prentice Hall, 1998

[34] K. Park and W. Willinger. Self-similar network traffic: An overview. Chapter 1 of Self-Similar Network Traffic and Performance Evaluation, Wiley-Interscience, 2000