Improving Performance of Online Game Services via Graphic Processor: An Empirical Investigation

Rittichai Jitpukdeebodintra[1] and Suntorn Witosurapot[2]

Department of Computer Engineering Faculty of Engineering, Prince of Songkla University
P.O. Box 2 Khohong, Hatyai, Songkhla, Thailand
[1]rittichai@capsuledna.com
[2]wsuntorn@coe.psu.ac.th

Abstract— **A method for maintaining quality of service in game servers with excessive users is often done by increasing the number of game servers. While this method is straightforward, it demands a number of new machines to be invested without realizing local utilization of resources available in the current machines. In this paper, we argue that graphic processor (GPU) working in parallel with local central processor (CPU) inside a machine can be a good candidate for reducing the workload, before attempting to distribute it to the other machines. By using the empirical study, we investigate in what level the GPU can give benefits to different types of online game servers. As a result, we can give suggestion how the GPU should be involved so that the performance of game services can be improved. We believe that this result can give benefits to online game developers who may want to gain performance of their applications without requiring any extra resources.**

Index Terms—**Online Game, Graphic processor utilization, GPGPU**

1. INTRODUCTION

MULTIPLAYER online game has been drastically gaining in strength and popularity in the game industry. To date, it becomes common that an online game may be served for hundreds or even thousands of concurrent players at same time. In this regards, the response time of multiplayer online games is crucial, especially during the high load of massive players. Hence, in order to lighten the load, the popular approach is often done by upgrading the current game server machine with a higher performance one. While it is easy and straightforward, this approach demands a new machine to be invested without utilizing some local resources, such as Graphic Processor Unit (GPU) and its memory, that are already available in the game server machine.

In this paper, we argue to take the GPU, which has shown an unbelievable excellent performance in many kinds of application, into consideration so that online games' service quality in term of response time can be maintained before attempting to distribute the excessive workload to the other machines. To support this argument, we describe our empirically evaluation and demonstrate our GPU-utilized online games. The contribution of this paper is in twofold; a) we advocate the GPU can be a good companion with the central processing unit (CPU) located in a machine for reducing the workload or accelerating some task during the heavy load, and b) we suggest on the different levels of GPU support for giving benefits to different service type of online game servers.

The remaining of the paper is organized as follows. In Section 2, we give a brief overview the GPU architecture, the difference of GPU and CPU architecture, and the programming features of new generation GPU. In Section 3, we describe in what way GPU can be involved into the online game server. In Section 4, we describe our experiments and results, followed by a suggested guideline on improving the performance of online game service via the GPU utilization. In Section 5 we conclude the paper.

2. BACKGROUND

A. Graphic Processor Unit Architecture

Due to the insatiably demanded high-definition 3D graphics in most modern games, the GPU hardware architecture has evolved into an enormous compute-intensive, highly parallel multithreaded multi-core processor [1], as illustrated in Figure 1 in comparison of general multi-core Central Processing Unit (CPU). As consequence, the GPU is then well-suited to speed up many computational applications, where a large data set can be processed via the mapping of data elements to parallel processing threads. In this case, the GPU can be referred as General Purpose computing on Graphic Processor Unit (GPGPU) technology. Hence, it is clear that the modern GPU architecture can potentially give benefits to most operations of game server machines, in order to maintain its service quality.

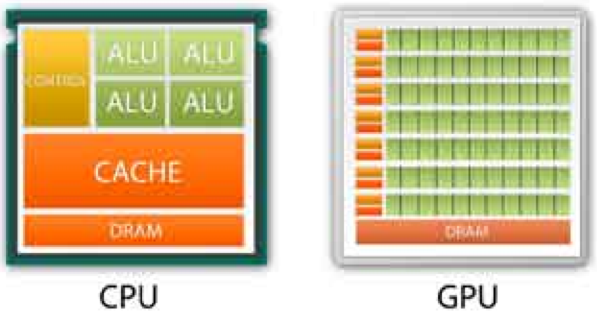


Fig. 1. CPU and GPU Architecture.

DOI: 10.5176_2010-2283_1.1.11

B. Related Works

Many research works have been found on improving the service quality of online game servers during the times of congestion. However, we are particularly interested in workload distributed techniques of online game server. For instance, the works in [2, 3] suggest how CPU resources of many computers in federation can be work in cooperation for serving as a unified infrastructure for hosting on-line games based on grid technology. In [4], it takes the similar approach to design a flexible and powerful infrastructure for Massive Multiplayer online (MMO) games, which cooperates among a number of servers working in the backend for different functionalities such as game server, Web application server, and database. Our work aims to achieve the same objective of CPU workload distribution, but focusing on the local GPU utilization for basic improvement, before attempting to distribute workload (i.e. game services) to the other machines of game servers. The concept can be easily seen in Figure 2.

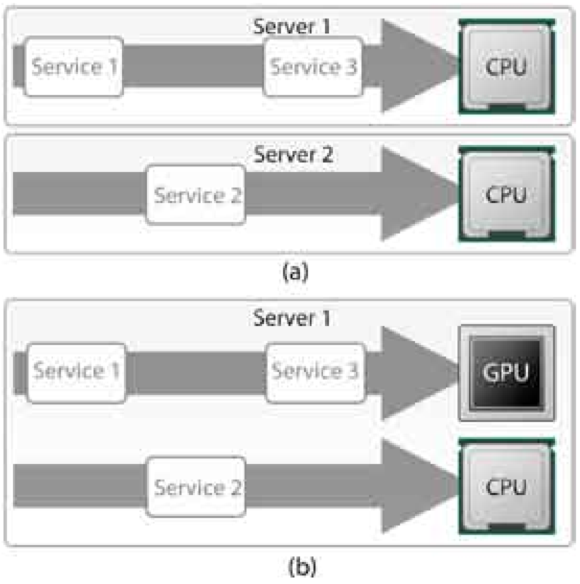


Fig. 2. Comparison of service distribution on game server machine: (a) General approach, and (b) Our proposed one.

3. PROPOSED MODEL

As mentioned previously in the Section 2, the GPU is so powerful that it can be simply used as a general-purpose GPU (GPGPU) for accelerating large data-parallel computations, but needs to work in collaboration with the local CPU on the same machine. However, in the context of online game server applications, it is doubtful in what level the GPU can give benefits to them. To explore this, we setup an collaboration model for further investigation in the next section. In Figure 3, the process flows occurring in our model can be described as in follows: When a client makes a request in (1) and the server should perform some computation and return a response accordingly (6).

1. In case of sole computing on the local CPU, all the client related parameters will be locally performed and kept in the main memory.
2. In case of distributed computing on the GPU, all the client related parameters will be copied to the GPU's memory and further executed under the command of CPU by means of the service wrapper functionality. When finished, the results will be later transferred to store on the main memory.

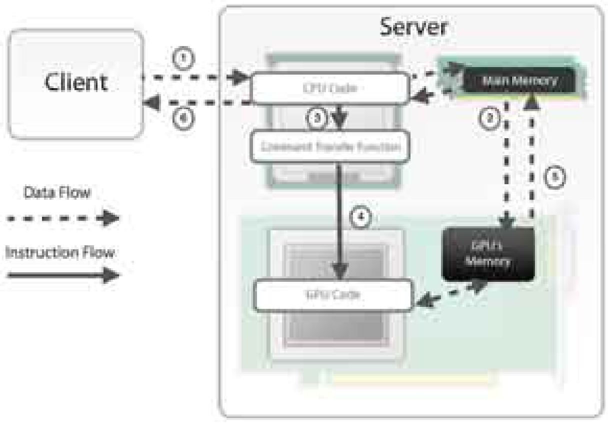


Fig. 3. GPU Function process

4. EMPIRICAL EVALUATION

We group the propose solution's empirical evaluation into 2 parts. 1 testing the propose solution in one player online game server and 2 testing the propose solution in massive multiplayer online games server. In this case, both parts using same experiment platform consist of 2.66 GHz Quad-Core Intel Core i7 (4 Core 8 threads CPU), 3GB DDR3 main memory, Windows 7 x86 with NVIDIA ForceWare version 190.38 as GPU driver and NVIDIA GeForce 9800GX2 graphics cards (Consists 256 Stream processors)

For the purpose of testing, we build two versions of each experimental online game; the version running solely on the CPU (Built with Microsoft Visual C# 2008) and the other version running on both CPU and GPU (Built with NVIDIA CUDA SDK 2.2). We repeat five times for each testing and then average the results to yield the final result showed in this paper.

A. Experiment 1: Single Player Online Game

1. The Model

We implement the Tic-tac-toe [10] program, where two players (one character is human and the other is computer (called a non-player character or NPC)) compete for placing three respective marks in a horizontal, vertical or diagonal row to win the game, as illustrated in Figure 4. However, two key services are particularly concerned in our developed program; a) AI-based service for computing NPC's next turn and b) Score calculation service for accumulating the user scores. It's noted that the AI-based algorithm can take direct advantage from the co-existed GPU by distributing the workload searching for the best strategy across GPU's parallel cores. This is not the case for the score calculation, where a single calculated operation is only processed.

In our experiments, either of these services will take different patterns for execution on both types of processors. The objectives are to observe in which context (or in what level) the GPU can give benefits to the execution of Tic-tac-toe game servers.

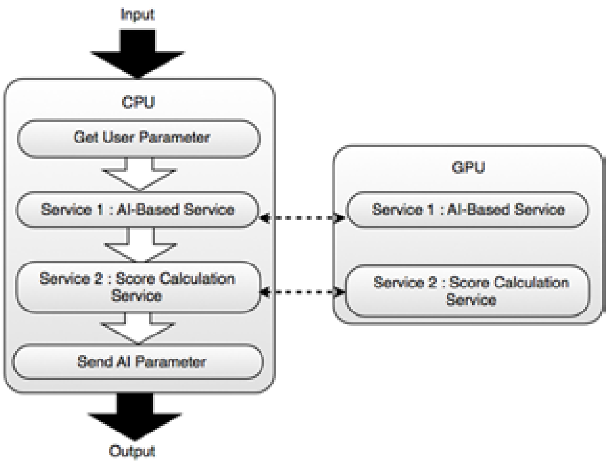

Input
CPU
Get User Parameter
Service 1 : AI-Based Service
Service 2 : Score Calculation Service
Send AI Parameter
GPU
Service 1 : AI-Based Service
Service 2 : Score Calculation Service
Output

Fig. 4. Workflow in Tic Tac Toe online game

2. Results

As we can see from the Table 1, the AI-based service running on the GPU noticeably outperforms (in term of producing overall response time) this service running on the CPU, while the Score calculation service is better running on the CPU, rather the GPU counterpart. This implies the shift of separable workloads to be executed on the GPU can potentially reduce the processing time.

TABLE I
OVERALL RESPONSE TIME OF SINGLE THREAD (TIC-TAC-TOE)

Test case number	AI-based service running on	Score calc. service running on	Elapsed Time (ms)
1	CPU	CPU	1.209
2	GPU	CPU	0.046
3	CPU	GPU	1.237
4	GPU	GPU	0.074

However, to ensure the better performance of overall system, we repeat the above scenarios, but increasing the number of inputs to 100 and 1000 clients. As the result shown in Figure 5, the overall CPU usage time in the Test Case number 2 is relatively shorter than that of the Test Case number 1.

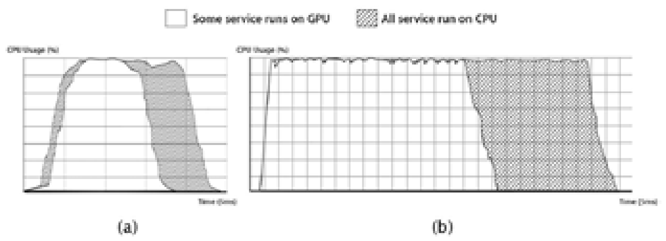


Fig. 5. Overall CPU loads comparison in 2 scenarios: (a) 100 workloads and (b) 1000 workloads

B. Massive Player Online Game

1. The Model

Here, we are interested in massive multiplayer online game server (like Final Fantasy XI Online, Ragnarok Online), but decide on the implementation of a massively Muliplayer online role-playing game (MMORPG) server called the virtual online stock exchange games. It is a kind of game that simulates the stock exchange environment as illustrated in Figure 6, and three key services are the main concerns in our developed program; a) Score ranking service for sorting all users' scores in the game server, b) Trader searching service for locating the stock trader in the database, and c) User status update service for updating all users' sores in the game server. Here, only the user status update service unlikely obtains the benefits from the co-existed GPU since a single calculated operation is only processed upon a request for a user. Here, each service will be individually performed by varying the number of input from 1 to 20000 clients and overall response time will be observed.

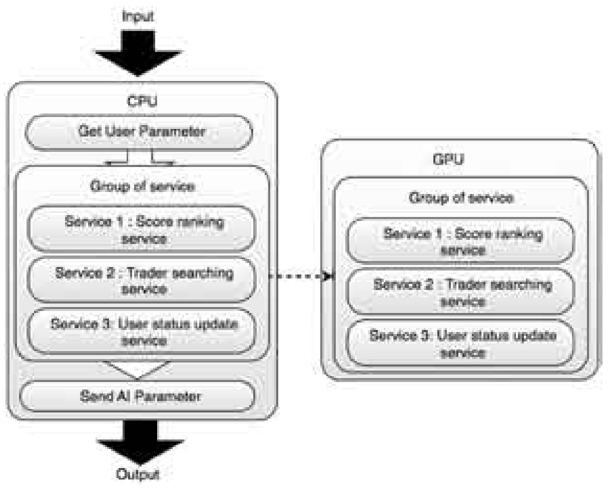
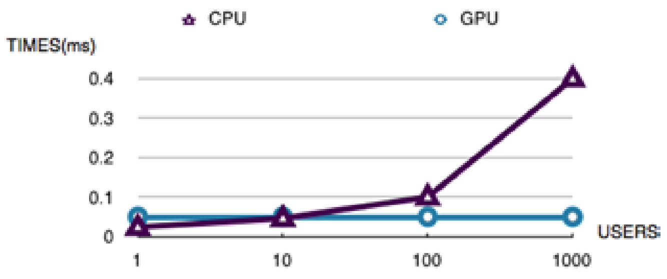

Input
CPU
Get User Parameter
Group of service
Service 1 : Score ranking service
Service 2 : Trader searching service
Service 3: User status update service
Send AI Parameter
GPU
Group of service
Service 1 : Score ranking service
Service 2 : Trader searching service
Service 3: User status update service
Output

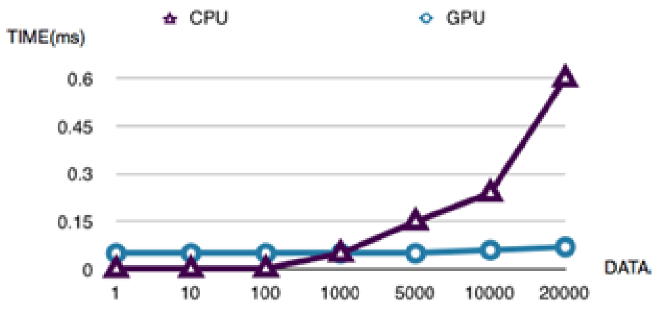
Fig. 6. Workflow in Virtual online stock exchange game

2. Results

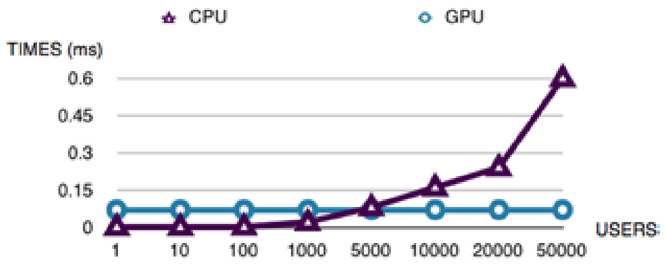
As seen in Figure 7, with a significantly large number of clients (approximately 2000 upward), the GPU can give the extreme benefits for all separable tasks such found in the searching trader service and the score ranking service. However, it is not the case when there is a small number of user data executed on the GPU due to the burden of associated overheads.



(a) Searching trader service



(b) Score ranking service



(c) User status update service

Fig. 7. Overall response time on various services

V. CONCLUSION

We have presented a possible approach to lighten CPU workloads of online game server application by favorable means of utilizing the local GPU computing facility. The objective is to express that the GPU can be a helpful factor for prolonging the fast response time to massive game players as long as the workload of game server machine is not overloaded. However, as clearly seen from our experimental results, it is not always the case that all sorts of online game will gain benefits owing to the cooperative works of these two processors. We then give suggestion on what sort of workload should be effectively executed on the GPU, therefore giving an extreme benefit not only to online game developers, but also the overall online game industry..

REFERENCES

[1] NVIDIA Corp 2009. NVIDIA CUDA Programming Guide 2.2 (2009)
[2] D Saha, S Sahu and A Shaikh 2003. A service platform for on-line games. *In Proceedings of Workshop on Network and System Support for Games* (2003)
[3] A Shaikh, S Sahu, M Rosu, M Shea and D Saha 2003. Implementation of a Service Platform for Online Games. *Proceedings of 3rd ACM SIGCOMM workshop* (2003)
[4] Hyun Sung Chu 2008. Building a simple yet powerful MMO game architecture (2008)
[5] S Caltagirone, M Keys, B Schlief and M Willshire 2002. Architecture for a massively multiplayer online role playing game engine. *Journal of Computing Sciences in Colleges* (2002)
[6] J Owens, D Luebke, N Govindaraju and M Harris 2007. *Computer Graphics Forum* (2007)
[7] J Glenn-Anderson 2009. GPU-based Desktop Supercomputing (2009)
[8] Gao Huang, Meng Ye and Long Cheng 2004. Modeling System Performance in MMORPG. *IEEE Communications Society Globecom 2004 Workshops* (2004)
[9] B Cowley, D Charles, M Black and R Hickey 2008. Toward an Understanding of Flow in Video Games. *Computers in Entertainment (CIE)* (2008)
[10] Tic-tac-toe, Wikipedia, the free encyclopedia (2010), http:// en.wikipedia.org/wiki/Tic-tac-toe

Rittichai Jitpukdeebodintra received his B.Eng. and M.Eng. in 2008 and 2010 respectively, both in Computer Engineering from Prince of Songkla University (PSU), Thailand. He is currently pursuing the Ph.D. degree at the department of Computer Engineering, Prince of Songkla University, Thailand. His research interests include Service-Oriented Architecture and Multimedia systems in the game industry.

Suntorn Witosurapot received his B.S. and M.S. degrees in 1982 and 1989, respectively from Prince of Songkla University (PSU), Thailand. He completed his Ph.D degree in Telecommunications from Swinburne University of Technology, Australia in 2005. Presently, he is an Assistant Professor at the department of Computer Engineering, Prince of Songkla University, Thailand. His current research interests are in the domain of distributed multimedia systems, middleware and Internet Mobile applications.