Proc. of The Sixth Intl. Conf. On Advances In Computing, Control And Networking - ACCN 2017

ISBN: 978-1-63248-117-7 doi: 10.15224/ 978-1-63248-117-7-56

# Load Balancing Scheme for Migration in Naval Combat System

[Hong-Keu Jo, Ji-Cheol Im, Hak-Hui Choi, Dong-Seong Kim*, and Jae-Min Lee]

***Abstract*—This paper proposes load balancing using live migration in a naval combat system on virtualization environment. When designing virtualization cluster for naval combat system, irregular load by the configuration node degrades performances. To solve this problem, this paper presents an algorithm for VM(Virtual Machine) which will be transferred in over loaded host, and improve stability and performance in the naval combat system through operate load balancing technology. Simulation results show that the proposed algorithm enhances the load distribution performance of the cluster.**

***Keywords*—Naval combat system, Virtualzation, Virtual machine, Live migration, Load Balancing.**

## I. Introduction

In future naval battle environment, battle ships require the ability to be assured of survival and to be able to successfully perform the assigned mission. Integrated management and automation functions of the sub-system in the naval combat system were required to meet the above requirement, and the research and development in the naval combat system based on the virtualization technology is in progress[1][2]. The virtualization system applied to the naval combat system virtualizes and aggregates the console and PC for the combat system in the physical server to increase energy efficiency, work efficiency and flexibility. At the same time, by increasing the operation efficiency of the server, it is possible to construct the same performance network at a lower cost than the existing physical system. However, when an unequal load occurs from the node(VM :Virtual Machine) in the host, the limit of the host resource is exceeded, so that the delay and error may occur from all the VMs included in the host, and the real time property cannot be guaranteed. This causes a great deal of problems in overall operational performance in the battlefield environment[3].

To solve this problem, there is live migration in virtualization environment. Live migration refers to the process of moving a running virtual machine or application between different physical machines(other host) without disconnecting to the client or application. In this process, since the VM application information and network connection state are transmitted, there is an advantage that the VM does not need to be stopped during the maintenance of the cluster and there is no need to make a reconnection request to the user. This increases the ease of maintaining virtualized environments and reduces the risk of service provision. In the case of non-virtualized environment migration, the active machine is stopped and then replicated, but the live migration performs the synchronization process after first performing the replication of the VM. Downtime is also needed during synchronization, but the time is several tens of *milliseconds*, which is shorter than the non-virtualized environments service downtime which have tens to hundreds *hours*.

But many existing studies proposes an algorithm to select a virtual machine to be migrated by monitoring the CPU usage of the virtual machine. If the CPU usage average over certain time exceeds the threshold on a host, a migration policy for load balancing is performed. At this time, migration is performed based on rank about CPU usage which is calculated with percentage. For example, the VM with the highest CPU usage is migrated to the host with the lowest CPU usage, or the VM which is above top 40% CPU usage is migrated to the host which have less than 50% CPU usage. The goal of migration is to ensure that idle resources are available to cope with overloads. But above scheme perform migration until the resource usage rank is maximized, so unnecessary migration is repeated even though the idle resource is sufficiently secured. In this case, an excessive migration attempt will cause unnecessary load on the cluster. Son et al.[4] presents an algorithm for load inequality resolution. In this paper, the system selects the VM with the most or least resource usage of the target VM. When selecting the VM with the highest resource usage, it is possible to greatly reduce the load even by one migration. However, the probability of finding a host having a large amount of idle resources is low, the failure probability of migration attempt is high. When selecting the VM with the optimal resource usage, the success probability of migration is high, but there is a disadvantage that the load reduction per migration is small. In addition, first, these papers do not include network traffic considerations and load prediction algorithms, It is possible to cause an operation delay due to network overload in the naval combat system where network resource management is important. And even though downtime of live-migration is shorter than normal migration, if unnecessary migration is repeated many times, it causes degradation of real-time performance and operational capability in the naval combat system. Therefore, an efficient algorithm is required to minimize the number of migrations in the load balancing process.

In this paper, we propose a cluster internal load balancing method using live migration to improve the high

Hong-Keu Jo
Kumoh national Institute of Technology
Republic of Korea

Ji-Cheol Im
Kumoh national Institute of Technology
Republic of Korea

Hak-Hui Choi
Kumoh national Institute of Technology
Republic of Korea

Dong-Seong Kim
Kumoh national Institute of Technology
Republic of Korea

Jae-Min Lee
ICT-CRC, Kumoh national Institute of Technology
Republic of Korea

Proc. of The Sixth Intl. Conf. On Advances In Computing, Control And Networking - ACCN 2017

ISBN: 978-1-63248-117-7 doi: 10.15224/ 978-1-63248-117-7-56

availability and stability of virtualized clusters. The proposed algorithm determines VM which will be transferred and destination host and migrates overload host and the VM that generates the overload to a low number of iterations.

# II. VM and Host selection algorithm

## A. *T(Type)-algorithm*

The host and VM in this paper consider resource information about C(CPU), M(Memory) and N (Network I/O). In order to define the relative load status among resources, it is necessary to compare three resources at the same time, but it is difficult to quantify them because they use different units. In addition, since VMs have different resource quantities, it is necessary to compare them by absolute values rather than relative values such as usage(%). To solve this problem, we define a scale value of virtual machine. The scale value of virtual machine has its value of resource quantity ( $_{sc}/C_{VM}$ , $_{sc}/M_{VM}$ , $_{sc}/N_{VM}$ ). The scale value of resource quantity of the VM is equal to the maximum value of $C_{VM}$ , $M_{VM}$ , $N_{VM}$ in the cluster as follows.

$$_{sc}/C_{vm} = \max(C_{VM}), \quad (1)$$

$$_{sc}/M_{vm} = \max(M_{VM}), \quad (2)$$

$$_{sc}/N_{vm} = \max(N_{VM}). \quad (3)$$

Likewise, scale value of host can also be defined. The scale value of host also has a scale value of resource quantity of the host, which is equal to the maximum value of $C_{Host}$ , $M_{Host}$ , $N_{Host}$ in the cluster. In this paper, since all the hosts in the cluster have the same hardware resources, the host resource value is equal to the total resource amount of the host as follows.

$$_{sc}/C_{Host} = \max(C_{Host}), \quad (4)$$

$$_{sc}/M_{Host} = \max(M_{Host}), \quad (5)$$

$$_{sc}/N_{Host} = \max(N_{Host}). \quad (6)$$

By defining as $_{sc}/C_{vm} = _{sc}/M_{vm} = _{sc}/N_{vm}$ , a comparison measure between different kinds of resources is created. For example, if scale value of virtual machine have $_{sc}/M_{vm}$ = 8Gb, $_{sc}/N_{vm}$ = 100Mbps, when VM1 have $M_{VM1}$ = 500Mb, $N_{VM1}$ = 50Mbps, it becomes $\frac{M_{VM1}}{_{sc}/M_{VM}}$ = 0.0625, $\frac{N_{VM1}}{_{sc}/N_{VM}}$ = 0.5, so that VM1 is judged to be a node having a relatively high utilization rate of network resources. By using the above values as a measure, virtual machines can

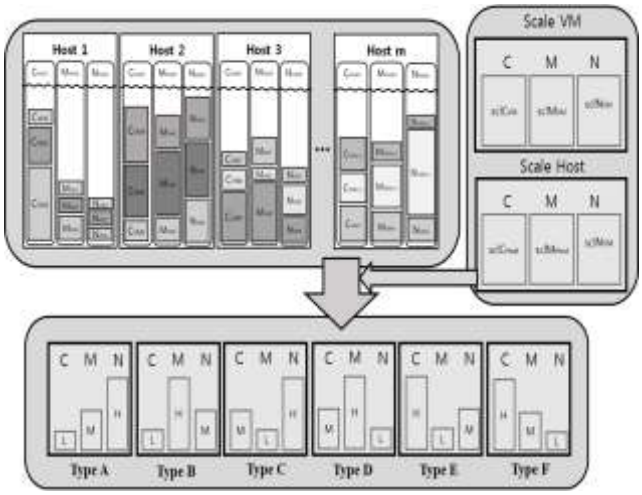


Figure 1. Conceptual diagram of T(Type)-algorithm.

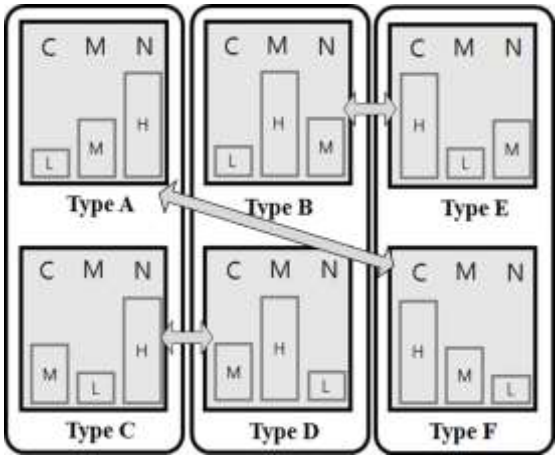


Figure 2. Similarity and symmetry relationships between types.

be defined in six types according to patterns of resource usage as shown in Fig. 1.

On the basis of the scale, all VMs and hosts are defined as 6 types when the highest used resource is H, medium is M, and L is low resource. The T-algorithm is for this type definition. By determining the type, it is easy to compare absolute values between the same types, and it is possible to determine the previous VM and the destination host in the migration process. There are 'similarity relation' and 'symmetric relation' between types. Fig. 2 shows the relationship between the six types and types of VMs. On the basis of the scale, all VMs and hosts are defined as 6 types when the highest used resource is H, medium is M, and L is low resource. The T-algorithm is for this type definition. By determining the type, it is easy to compare absolute values between the same types, and it is possible to determine the previous VM and the destination host in the migration process. There are 'similarity relation' and 'symmetric relation' between types. Fig. 2 shows the relationship between the six types and types of VMs.

## B. *R(rank)-algorithm*

After defining the type of host and VM with T-algorithm, we will prioritize the VM which to be migrated and the host

Proc. of The Sixth Intl. Conf. On Advances In Computing, Control And Networking - ACCN 2017

ISBN: 978-1-63248-117-7 doi: 10.15224/ 978-1-63248-117-7-56

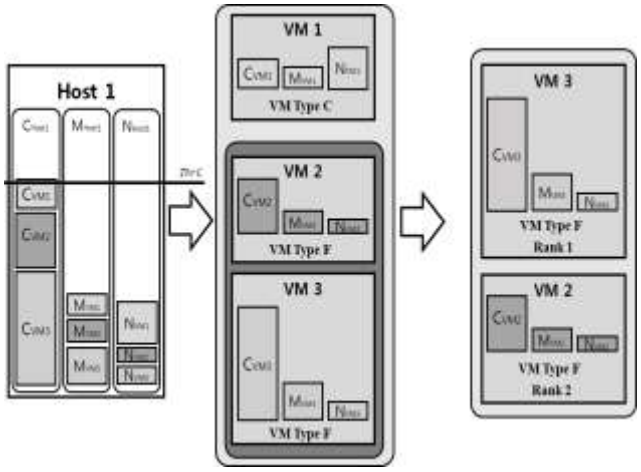


Figure 3. Conceptual diagram R(Rank)-algorithm.

which to be the migration destination. Fig. 3 shows the operation of R(Rank)-algorithm for this task. In Fig. 3, Host 1 is defined as an overload host exceeding the CPU resource threshold(ThrC). Thereafter, the VMs in the host1 are classified through the T-algorithm. In order to reduce the CPU resource amount, which is an overload resource. VM 2 and 3, which are VMs of the same type F as the host 1, are selected. Two VMs of the same type can have a clear comparison of the absolute resource usage value, and a high rank is given to the VM with the highest resource usage. The VM with the highest rank is selected as the prior target VM, and as a result, the VM3 having the highest resource usage among the type F VMs is selected as the transfer target VM. The process of selecting a destination host is the same as above. In order to accommodate Type F VMs with high probability, Type A hosts that are symmetrical are selected from the inside of the cluster, and a high rank is given to hosts having the largest idle resources. ////Unlike this, if the host includes a Similarity relation VM, as shown in Fig. 4, the system also assigns a rank to the similarity relation VM. As shown in Fig. 4, when there is a similarity relation virtual machine VM1, a rank is assigned to similarity relation virtual machine. If the migration of the F-type virtual machine fails or an F-type virtual machine does not exist. Similarity relation virtual machine is selected as the target virtual machine according to this rank.

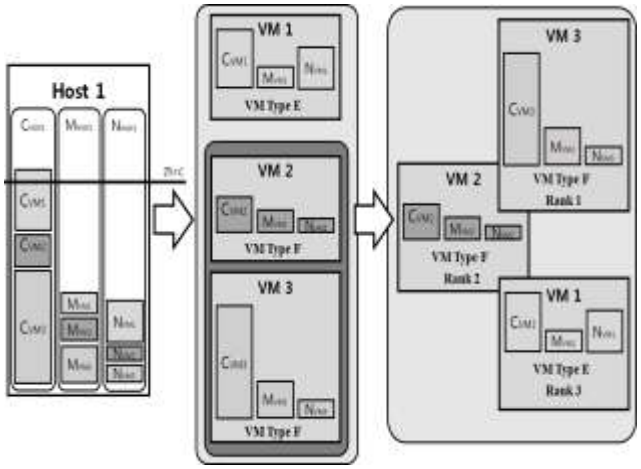


Figure 4. R(Rank)-algorithm in the host which includes similarity VMs.

TABLE I. SIMULATION ENVIRONMENT

| Specification | | | |
|---|---|---|---|
| ***Ethernet Switch*** | 2Gbps | ***Observation time*** | t = 120 [sec] |
| ***LAN Cable*** | CAT 7 Cable (600Mbps bandwidth) | ***Observation cycle*** | p = 15 [sec] |
| ***Host*** | processor (I5M Cache, 2.40 GHz, 24Cores) memory (DDR4-1600MHz 32G Memory) 1TB HDD, 240GB SSD | ***Host number*** | m = 5 |
| ***NAS*** | Processor(I3-3240, 3.40 GHz 2Cores) memory (DDR3 4.00GB) 750GB HDD | ***VM number*** | n = 100 |
| ***VM*** | Processor(I3-3240, 2.40 GHz) (1 or 2 or 4Cores) Memory (DDR3), (2 or 4 or 8 GHz) 30GB HDD | ***ThrC ThrM ThrN*** | 80%($C_{HostM}$, $M_{HostM}$, $N_{HostM}$)* 0.8 |

Likewise, if there is no suitable type of host in the destination host selection process, the similarity relation host is designated as the destination host according to the rank.

# III. Performance Analysis

Table I shows the specifications of the simulation nodes for performance analysis of the proposed load balancing scheme. Fig. 5 shows the number of migration decision and the number of attempts for each algorithm. An imbalanced based algorithm represents the algorithm in existing study[5].

In the case of random migration, the number of attempts was relatively high, because made a random migration decision. During the observation period, the number of migrations of the proposed algorithm is 6 and the number of migrations of the imbalanced based algorithm is 14. The imbalanced based algorithm has been shown to repeat unnecessary migration.

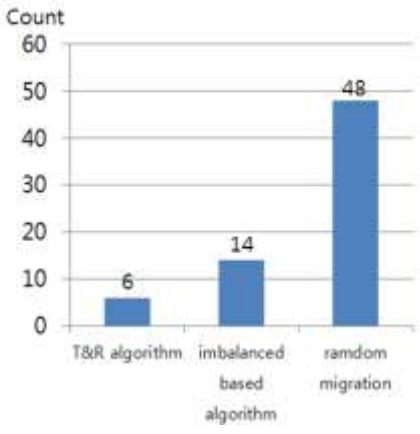


Figure 5. Number of migration attempts per algorithm.

Proc. of The Sixth Intl. Conf. On Advances In Computing, Control And Networking - ACCN 2017

ISBN: 978-1-63248-117-7 doi: 10.15224/ 978-1-63248-117-7-56

# IV. Conclusions & Future work

In this paper, we consider the network resources unlike the existing migration method and propose the T&R algorithm to overcome the overload and system operation delay. Through the simulation result, the proposed algorithm shows better load balancing performance than existing algorithms and improves the probability of successful migration. In future work, we will study the effectiveness of the proposed algorithm in a large battleship environment. The proposed algorithm will be implemented to existing virtualization platforms to enable advanced load balancing.

## *Acknowledgment*

This research was financially supported by National Research Foundation of Korea (NRF) through the Human Resource Training Project for Regional Innovation 2015 (NO. NRF-2015H1C1A1035971), the MSIP(Ministry of Science, ICT and Future Planning), Korea, under the "Creative ICT Convergence Human Resource Development Program" (NO. IITP-2017-H8601-15-1011) support program supervised by the IITP(Institute for Information and communications Technology Promotion) and BK21+ Project in Kumoh National Institute of Technology.

## *References*

[1] Jin-Yong Im, Dong-Seong Kim, "Performance Evaluation of Commercial Virtualization Scheme for Next Generation Naval Combat System," 2016 KICS Summer Conference, South Korea, vol. 60, pp. 368–369, 2016.

[2] Yang Wei, Dong-Seong Kim, "Enhanced Network Recovery Scheme on Real-Time Switched Ethernet for Naval Combat System," International Journal of Communication Networks and Distributed Systems, vol. 14, No. 2, pp. 145-163, 2015.2.

[3] Dong-Hee Noh, Dong-Seong Kim, "Markov Model-Driven in Real-time Faulty Node Detection for Naval Distributed Control Networked Systems," ICROS, vol. 20, No.11, pp. 1131-1135, 2014.11.

[4] Sung-Hoon Son, Sang-Il, "Load Balancing Policy for Xen-based Virtual Desktop Service", KCI, South Korea, Vol. 8, No. 6, pp. 916–929, 2015

[5] J. Octavio Gutierrez-Garcia, Adrian Ramirez-Nafarrate, "Collaborative Agents for Distributed Load Management in Cloud Data Centers Using Live Migration of Virtual Machines", IEEE Transactions on Services Computing, Vol. 13, No. 1, pp. 55–64, 2008. 1.

He is a Master student at IT Convergence Department, Kumoh National Institute of Technology, South Korea. His research interests are wireless sensor networks, virtualization and embedded systems.

He is a Master student at IT Convergence Department, Kumoh National Institute of Technology, South Korea. His research interests are wireless sensor networks, virtualization and embedded systems.

He is a Master student at IT Convergence Department, Kumoh National Institute of Technology, South Korea. His research interests are wireless sensor networks and embedded systems.

Dong-Seong Kim received his Ph.D. degree in Electrical and Computer Engineering from the Seoul National University, Seoul, Korea, in 2003. From 1994 to 2003, he worked as a full-time researcher in ERC-ACI at Seoul National University, Seoul, Korea. From March 2003 to February 2005, he worked as a postdoctoral researcher at the Wireless Network Laboratory in the School of Electrical and Computer Engineering at Cornell University, NY. He is currently a director of kit Convergence Research Institute and ICT Convergence Research Center supported by Korean government at Kumoh National Institute of Technology. He is IEEE and ACM senior member. His current main research interests are industrial wireless control network, networked embedded system and Fieldbus.

Jae-Min Lee received his Ph.D degree in in Electrical and Computer Engineering from the Seoul National University, Seoul, Korea, in 2005. From 2004 to 2016, he worked at Samsung Electronics Co., Ltd. as a researcher. He is currently a Prof. of ICT Convergence Research Center, Kumoh National Institute of Technology, South Korea. His research interests are real-time network, analysis of wireless network and TRIZ.