Making VM Consolidation More Energy-efficient by Postcopy Live Migration

Takahiro Hirofuchi, Hidemoto Nakada, Satoshi Itoh, and Satoshi Sekiguchi
National Institute of Advanced Industrial Science and Technology (AIST)
Central 2, Umezono 1-1-1, Tsukuba, Japan 305-8568
Email: t.hirofuchi@aist.go.jp, hide-nakada@aist.go.jp, satoshi.itoh@aist.go.jp, s.sekiguchi@aist.go.jp

Abstract—**Dynamic consolidation of virtual machines (VMs) is a promising technology for reducing energy consumption of data centers. Existing studies on VM consolidation, however, are based on** *precopy live migration*; **it is difficult to optimize VM locations aggressively due to its long and undeterminable migration process. In this paper, we propose an energy-efficient VM consolidation system exploiting** *postcopy live migration*, **which always allows quick live migration for any VMs. The consolidation system can optimize VM locations and server power states more frequently than those of using precopy live migration. In our previous work, we implemented postcopy live migration for KVM, and in this paper, we developed the prototype of our consolidation system, where excessive hardware nodes were suspended by means of ACPI S3 and all power usages were monitored with watt meters. Our experiments showed that our consolidation system with postcopy live migration eliminated more excessive power consumption than that of using precopy live migration. Postcopy live migration allowed the prototype system to eliminate 11.8% energy overheads of actively-running VMs, which was improved by approximately 50% from precopy live migration.**

***Keywords*-Virtual Machine; Live Migration; Consolidation; Data Center; Energy Saving.**

I. INTRODUCTION

Dynamic consolidation of virtual machines (VMs) is a promising technology for reducing energy consumption of data centers. The number of power-on server nodes is kept to a minimum at any time, so that the excessive power used for running idle server nodes can be eliminated. The locations of VMs are continuously reoptimized in response to resource requirements of VMs. When there are many idle VMs, a management system consolidates them onto fewer server nodes, and temporarily shuts down the rest of the server nodes. When these idle VMs become active, the system wakes up power-off server nodes, and relocates VMs onto them.

Live migration of VMs greatly contributes to realizing dynamic consolidation. A VM is relocated onto a new server node without any visible disruption. It should be noted that power consumption incurred by live migration itself is a relatively small value, compared to power saving gains by consolidation. As discussed in Section II, in our experiment environment, making an idle server to the suspend state of ACPI reduces 40W and more, and the network traffic and CPU overhead of a live migration consumes approximately only 7W. This means that a management system is required to perform live migrations as many times as possible in order to maximize the energy-efficiency of VM consolidation.

Widely-used live migration mechanisms, however, are not suitable for dynamic consolidation, which cannot relocate VMs frequently due to their long migration duration. These live migration mechanisms are known as *precopy* live migration; all states of a VM are completely copied to a destination node before the execution host is switched to the destination. Until the whole migration process is completed, the VM is still running on a source node. Updated memory pages during previous page copies are repeatedly transferred to the destination. This iteration process results in a long and undeterminable migration time for actively-running VMs.

On the other hand, there are also *postcopy* live migration mechanisms, performing memory page copies after the execution host is switched. This migration does not need iterative memory copies. A migrating VM updates memory pages at a destination node, not at a source node, which do not generate additional data to be transferred. The total amount of transferred data is smaller than precopy; the whole live migration process is shorter and determinable.

We believe postcopy live migration enables more energy-efficient VM consolidation than precopy live migration. To the best of our knowledge, however, all existing studies on VM consolidation are based on precopy live migration. There are open questions regarding how postcopy live migration contributes to power savings of data centers.

In this paper, we propose an energy-efficient VM consolidation system exploiting postcopy live migration. Postcopy live migration enables the consolidation system to aggressively control VM locations and server power states. The proposed system achieves more frequent live migrations and server power state changes. This fine-grained optimization allows the system to eliminate excessive energy consumption much more than using precopy live migration.

The contribution of this paper is clear; this study is the first work that applies postcopy live migration to an energy-saving VM consolidation system. Although postcopy migration techniques themselves have been discussed in research papers ([1], [2]), these implementations have not been seen in publicly-available VMMs. We therefore developed a postcopy live migration mechanism [3] for KVM [4]. In our previous work [5], we discussed the advantages of our postcopy live migration from the viewpoint of performance assurance for VM consolidation. In this paper, we address the remaining questions of how much energy savings are possible with our postcopy live migration. We have developed a consolidation system using the ACPI S3 mode and evaluated the effectiveness of postcopy live migration through various experiments.

Section II explains how VM consolidation systems basically work, and summarizes why postcopy live migration has great advantages for energy savings. Section III presents our VM consolidation system. Section IV discusses its evaluation. Section V describes related work. Finally, Section VI concludes this paper.

II. Background

Energy saving technologies are keys to success in the data center business, which allow service providers to reduce daily running costs. The recent processors technologies, such as Dynamic Voltage and Frequency Scaling (DVFS) and ACPI C State [6], contribute to reducing energy consumption of running server nodes. However, these technologies cannot cut the excessive power usage of other hardware components, such as a power supply unit and a mainboard. A study on a large data center mentioned servers were operating most of the time at between 10 and 50 percent of their maximum utilization levels; however, the energy efficiency of server hardware in these utilization levels is less than half at peak performance [7]. Although recent data center facilities, such as direct current power supply systems, mitigate this issue, the deployment of these technologies requires large modifications to existing server platforms and facilities. This results in high implementation costs in most data centers.

Emerging virtualization technologies allow VM-based server consolidation for data centers. A consolidation system monitors resource usage of VMs and continuously optimizes VM locations. The system packs VMs onto the fewest possible server nodes and powers off unused server nodes. When detecting the overloading of a server node, the system powers up unused server nodes and relocates some VMs onto them. Even though most VMs are operating at lower utilization levels, the utilization levels of power-on server nodes are always kept high by packing all VMs onto them. Ideally, the energy consumption of all server nodes is proportional to the total resource usage of all VMs. VM consolidation allows service providers to eliminate excessive energy consumption that are not used for customers' computations.

Figure 1 illustrates the overview of our consolidation system. Load Monitor collects resource usage data every one second and put it into a database. Relocation Planner periodically calculates optimal locations for VMs from the latest resource usage histories in the database. VM Controller requests live migration to server nodes according to the results from Relocation Planner. Although consolidation systems

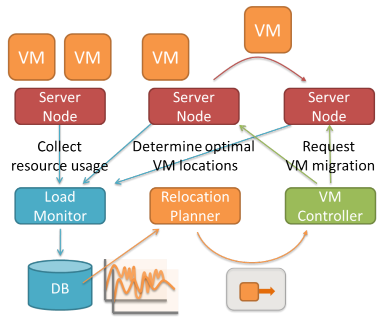

Figure 1. System components of our consolidation system

Table I
Specification of Server Node and Network Switch

Server Node	Dell Optiplex 960 CPU: Intel Core2 Q9400, RAM: DDR3 16GB HDD: ST380815AS Seagate 80GB GbE NIC: Intel 82567LM-3 GbE NIC: Broadcom NetXtreme BCM5721
Network Switch	Planex FXG-24IRM (GbE, 24 port)

have different design details, the above system overview is basically common to most consolidation systems.

Next, we explain the energy consumption breakdown of our VM consolidation system, and then point out why postcopy live migration is suitable for VM consolidation.

A. Energy Consumption Breakdown of VM Consolidation

1) Power Consumption of a Server Node: Before discussing requirements for energy-efficient VM consolidation, we measured power consumption of a server node in our cluster. The specification of the server node is summarized in Table I.

We use a customized Dell Optiplex 960, which supports the ACPI S3 mode and an out-of-band hardware management system (Intel AMT) [8]. Our consolidation system requires a hardware mechanism that allows VM Controller to wake server nodes up via a network. We first tried to use the Wake-On-LAN (WOL) feature, which is widely supported by most network interface cards. However, we found that WOL was not reliable enough to be used in a server cluster. The WOL message is transferred by a UDP datagram, which is likely dropped in congested networks. In practice, if the consolidation system is deployed on a large server cluster, each server node also needs to support more powerful remote hardware management than WOL; the hardware and software settings of all server nodes should be reconfigurable from a remote administrative program. Intel AMT (Active Management Technology), working in the firmware level, allows powerful remote management including power status control, console redirection, and OS installation. Intel AMT exploits TCP connections for its RPCs, making remote control more reliable than other UDP-based remote management mechanisms (e.g., IPMI [9]).

Table II
SERVER ENERGY CONSUMPTION (POWER ON)

CPU Usage (%)	C-State	Watt
100	Enabled	100
100	Disabled	100
0	Enabled	53
0	Disabled	64

Table III
SERVER ENERGY CONSUMPTION (POWER OFF/SUSPENDED)

State	Intel AMT	Watt
Power Off	Enabled	6
Power Off	Disabled	0
Suspended	Enabled	7
Suspended	Disabled	7

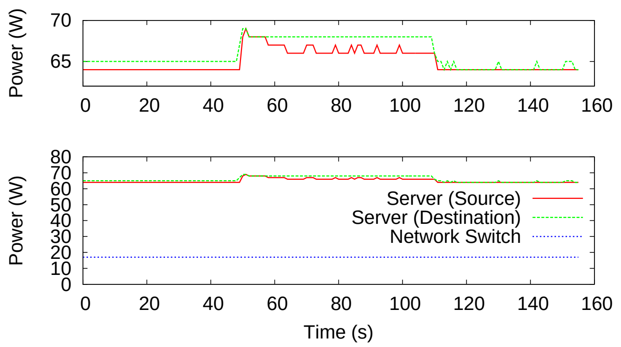

Figure 2. Energy consumption of server nodes and a network switch (A live migration is performed from 45 seconds to 105 seconds. The upper graph shows details around 65W.)

Tables II and III show energy consumption of a server node in its various states and settings. The server node, running at its full CPU utilization, consumes approximately 100W. The idle server node consumes 64W without the power saving feature. The ACPI C State, which enables an idle CPU to stop its clock cycle, contributes to reducing only 11W. Even though the server node is idle, it still consumes approximately half of the power usage at the maximum utilization level. It should be noted, DVFS (i.e., scaling up/down CPU's clock frequency), cannot reduce idle CPU power additionally; the clock cycle is already stopped by the C State feature.

When the server node is suspended, its power consumption is only 7W; this is a much smaller value than an idle power-on state. An interesting finding is that when Intel AMT is enabled the powered-off sever node still consumes 7W. Even though an operating system has been shut down, the firmware OS of Intel AMT is still running. When Intel AMT is disabled, the power consumption is approximately 0W. However, as mentioned before, this feature is required to control server power states remotely.

The results are summarized as follows: First, the contribution of CPU's power saving features is much smaller than making a server node shutdown. Second, because the recent out-of-band management technology, Intel AMT, requires its firmware OS to keep always running, the power consumption in the power-off state is not zero; in our experiments, it is approximately 7W, which is equal to the suspended state.

2) Power Consumption of a Live Migration: Figure 2 shows energy consumption of server nodes and a network switch when a live migration was performed in our experiment environment. An idle VM with 2GB memory was migrated between two server nodes via a GbE network. The normal live migration mechanism of KVM was used. It took approximately 60 seconds to be completed. While the live migration was being performed, the power consumption of each server node increased by 3W or less; this was mainly caused by the CPU overhead of the live migration. Although more than 2GB data was transferred via the network switch, its energy consumption did not show a visible increase. It should be noted that the power consumption of the network switch is nearly invariable while being powered on; the power consumption does not depend on how much data is being transferred now.

In our experiment environment, the additional power consumption incurred by a live migration is approximately 0.08Wh, which is calculated by integrating the power increase during the migration period. This value is much smaller than that of continuing to run a server node; 0.08Wh is corresponding to the power consumption of running an idle server node only in 5 seconds.

B. Requirements for Energy-Efficient VM Consolidation

These results have pointed out design criteria for energy-saving VM consolidation. First, to get the maximum energy saving, a VM consolidation system should exploit the ACPI S3 feature to turn off idle server nodes. The amounts of power consumption at the S3 state and power-off state are the same in our experiment environment. By using the ACPI S3 feature, the consolidation system can turn off/on a server node only in 5 seconds or less. This is much shorter than powering off/on the server node. To power off the server node, it takes approximately 20 seconds after the `shutdown` command is invoked. After the power-on command is invoked via Intel AMT, the VMM on it becomes operational approximately in 60 seconds. These long transitional periods result in increasing excessive power usage, which is not consumed by actual computations of VMs.

Second, the VM consolidation system should repack VMs as aggressively as possible to make excessive server nodes temporarily sleep. As discussed previously, at the viewpoint of power consumption, the overhead of a live migration is far less than that of continuing to run an excessive node; although the power consumption during the transition period of a suspend (e.g., 5 seconds) is also considered, the repacking overhead with one migration and one node suspend incurs only 0.15Wh (i.e., corresponding to approximately 10 seconds power consumption of an idle node). This means, to get the maximum energy saving, the consolidation system should be designed to be able to optimize VM locations at shorter intervals than one minute. Existing studies concerning VM packing have not addressed this kind of frequent optimization at such short intervals. On the other hand, we aim to establish fine-grained, aggressive optimization at the level of every 10 seconds, not in daily and weekly cycles.

C. Limitation of Precopy Live Migration

Prior studies regarding VM consolidation are based on precopy live migration, which is already available in widely-used VMMs (e.g., Xen [10], KVM, and VMware [11]). We believe, however, precopy live migration is not suitable for energy-efficient VM consolidation, because of its undeterminable (and possibly large) migration time.

It reconstructs a VM's memory image at a destination host **before** switching its execution node ([12], [13], [14]). After live migration is initiated, this basically works as follows.

1: Start dirty page logging at a source host. This mechanism detects updates of memory pages during the following memory copy steps. **2:** Copy all memory pages to the destination. Since the VM is running at the source host, memory pages are being updated during this period. **3:** Copy dirtied memory pages to the destination again. Repeat this step until the number of remaining memory pages is small enough. **4:** Stop the VM at the source. Copy the content of virtual CPU registers, the states of devices, and the rest of the memory pages. **5:** Resume the VM at the destination host.

The problem of precopy live migration is caused by the third step; dirtied pages must be iteratively copied to the destination again and again. If the VM is running a memory-update-intensive workload, numerous dirty pages are created and transferred continuously. The total time of precopy live migration basically becomes much larger than that of *cold migration* (i.e., stop the VM, send its state to a destination, and restart the VM). In the worst case, live migration is never completed; i.e., a workload dirties VM memory faster than network bandwidth can accommodate.

This large migration time prevents a consolidation system to optimize VM locations frequently. It is not possible to maximize energy efficiency of VM consolidation.

III. Energy-Efficient VM Consolidation with Postcopy Live Migration

We propose an energy-efficient VM consolidation system exploiting postcopy live migration. In this section, we explain the advantage of using postcopy live migration, and describe the design and implementation of our VM consolidation system.

A. Postcopy Live Migration

In previous work [3], we developed a postcopy live migration mechanism for KVM. In contrast with precopy migration, memory pages are transferred **after** a VM is resumed at a destination host. The key to postcopy migration is an on-demand memory transfer mechanism, which traps the first access to a memory page at the destination and copies its content from a source host. Postcopy migration basically works as follows:

1: Stop the VM at the source host. Copy the content of virtual CPU registers and the states of devices to the destination. **2:** Resume the VM at the destination without any memory content. **3:** If the VM touches a not-yet-transferred memory page, stop the VM temporarily. Copy the content of the memory page from the source. Then, resume the VM.

The third step is repeated until all memory pages are transferred to the destination. In addition, in parallel with the on-demand page retrievals, a background copy mechanism works to make bulk copies of not-yet-transferred pages. Because on-demand page copy may not cover all ranges of VM memory in a short period of time, the background copy mechanism gets rid of dependency on a source host as soon as possible. The background copy mechanism analyzes important memory areas with page fault statistics, and starts to deal with hot-spot memory pages for current VM workloads. On-demand memory page retrievals over a network are reduced by this mechanism. These mechanisms are transparent to the users of the VM. Our postcopy live migration mechanism supports any guest operating systems without any modifications to them.

A postcopy live migration is always completed in $Ramsize/Bandwidth$ seconds, which is much shorter than precopy. On the other hand, a precopy live migration requires $Ramsize/Bandwidth + \alpha$ seconds to be completed. α depends on the memory update speed of the guest operating system; if a VM intensively updates memory or a network is congested, α becomes larger, and in the worst case the migration is never completed.

The possible downside of postcopy migration is the risk to failure of VMs. A migrating VM depends on not-yet-transferred memory pages on its source host. If the source host is unexpectedly terminated during the migration, the VM cannot continue running anymore. However, considering that IaaS data centers do not assure 100% reliability of their services, we believe that this trivial downside does not adversely affect the feasibility of postcopy migration. As explained in the later sections, postcopy migration greatly improves energy efficiency of dynamic consolidation, which results in great benefits for service providers.

B. VM Consolidation System

Figure 3 shows the design overview of our VM consolidation system. Broadly speaking, there are 3 types of physical

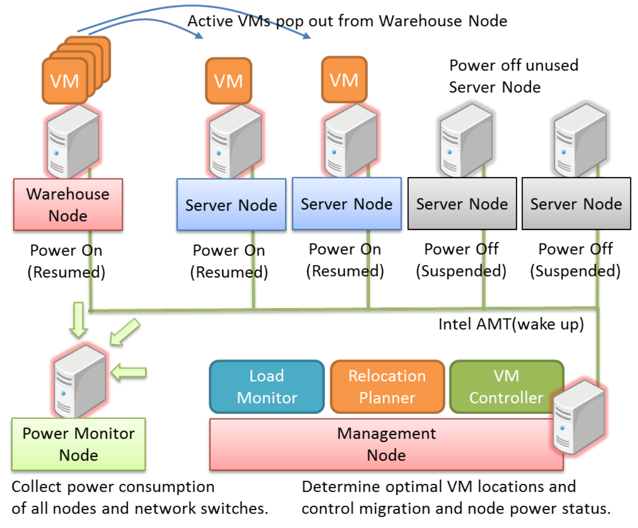

Figure 3. Design overview of our consolidation system

nodes in our server cluster. **Management Node** periodically determines optimal VM locations and controls migration and node power status, where the aforementioned software components are running (See also SectionII). **Power Monitor Node** collects power consumption of all nodes and network switches. **Host Nodes** (i.e., Warehouse/Server Nodes) launch VMs and execute live migrations of VMs.

1) Management Node: Load Monitor receives resource usage statistics from each host node, such as CPU usage, network I/O, and disk I/O of both a host node and the VMs running on it. This information is retrieved from `/proc/` of the host Linux operating system and the monitor interface of KVM. All the collected statistics are stored in an SQLite database. In order to support hundreds of host nodes, the latest statistics are temporarily cached in the memory of Load Monitor, thereby reducing database requests.

Relocation Planner retrieves resource usage histories from the database, determines whether a host node is overloaded or not, and calculates a relocation plan. We carefully designed this component to be independent from the others, so that it is possible to implement various consolidation algorithms.

VM Controller executes live migration according to the relocation plan. We use XML-RPC to control VMs on host nodes remotely; three request messages (e.g., `CREATE_VM`, `MIGRATE_VM`, and `DESTROY_VM`) are defined to create, migrate, and destroy the requested VM. On each host node, there is a server daemon handling these XML-RPC requests. VM Controller also executes the suspend/resume of host nodes. When all VMs on a host node are removed from it, VM Controller requests the host operating system of the node to invoke the `pm-suspend` command. When a suspended host node is required to run a VM, VM Controller requests the firmware of the host node to wake it up via Intel AMT.

Figure 4. Power Measuring System

2) Power Monitor Node: We developed a power measuring system of our server cluster, which periodically collects power consumption of host nodes and network switches individually. The current, voltage, and active power of a target component are measured by a customized watt meter; we use Watt Checker (MWC-01) of Osaki Electric Co, Ltd. The accuracy of active power is $\pm 2\%$. The measurement interval of the watt meter is one second. All watt meters are connected to a monitoring server (i.e., Power Monitor Node) via USB interfaces. It is possible to measure power consumption of 120 target components. Figure 4 is a photo of a part of our power measuring system; a 2U rackmount measuring board for 8 target components is installed into a 19-inch rack.

3) Host Nodes (Warehouse and Server Nodes): To consolidate VMs efficiently, our consolidation system introduces two types of host nodes, Server Nodes and Warehouse Nodes. Actively-running VMs are assigned onto Server Nodes, and idle VMs are packed into Warehouse Nodes. If a VM running at a Server Node becomes idle (i.e., consuming few CPU resources), the system migrates the VM to a Warehouse Node, and suspends the Server Node if there are no VMs anymore.

This design choice is made by considering hardware costs and use cases. To pack idle VMs into the minimum host nodes, the system should have a special host server with a large amount of physical memory; a Warehouse Node is dedicated to hosting as many idle VMs as possible. On the other hand, Server Nodes have a small amount of memory to host a few active VMs. Because active VMs will make substantial impacts on CPU and I/O resources, these VMs should be located on other nodes than Warehouse Nodes.

C. Packing Algorithm

Our consolidation system is designed to be independent of packing algorithms. It is possible to implement any kinds of packing algorithms. In the first prototype system, we implemented a simple heuristic algorithm that determines near-optimal locations swiftly. An active VM is exclusively assigned to a Server Node; on the other hand, idle VMs share a Warehouse Node.

First, all the VMs are launched at one of the Warehouse Nodes, and then the following steps are iterated every second.

Distribution Phase: When the latest 10-seconds CPU load average of a Warehouse Node reaches 90% (i.e., is regarded as overloaded), the most CPU-consuming VM is migrated to a Server Node. By using usage statistics measured in outside of VMs, it is difficult to accurately determine the amount of a CPU resource is actually required. Therefore, simply, we pick up the VM that is probably in a 'race-to-halt' state. A target Server Node is selected from sleeping Server Nodes, and then resumed to accept the VM. The VM is migrated to the Server Node. Finally, if there are no VMs on the Warehouse Node, the consolidation system suspends it.

Consolidation Phase: The system does not move the migrated VM for at least 20 seconds after the migration ends, in order to avoid overreaction. After that, the resource monitoring daemon of the VM is started to periodically check whether the latest CPU load average of the VM is under a return threshold value (50%). If the load average is under the threshold, the monitoring daemon tries to move the VM back to one of the Warehouse Nodes; it tries to find the Warehouse Node that has sufficient CPU and memory resources for the VM. If the Warehouse Node is suspended, the consolidation system resumes it. An *admission ticket* to a Warehouse Node is given to the VM on a 'first come, first served' basis, in order to serialize migrations to the Warehouse Node. If a Warehouse Node with sufficient resources is found, the VM is migrated to it. Otherwise, the VM remains at the Server Node; the daemon pauses at one second intervals and tries the above steps again.

It should be noted that the algorithm is currently based on only CPU usage statistics, not including disk and network I/O data. At the time this paper is being written, KVM does not support live migration for paravirtualized devices, such as VirtIO Block Device and VirtIO Network Device. All the VMs on our consolidation system must use fully-virtualized devices incurring relatively high CPU overheads.

IV. Evaluation

In our testbed cluster, we performed experiments to evaluate the effectiveness of our consolidation system; our consolidation system with postcopy live migration was compared with that of using precopy live migration. We measured energy consumption of our consolidation system with simple and complex workload scenarios.

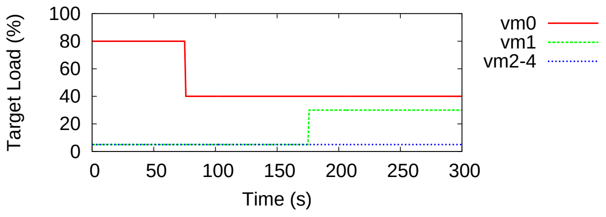

Figure 5. The CPU Load Changes of VMs in a Simple Consolidation Scenario

A. Experiment Settings

Our testbed cluster includes 6 host nodes of the specification in Table I; one node is used for a Warehouse Node, and other 5 nodes are used for Server Nodes. Each host node is connected to a shared disk server, which is required to perform live migrations among different host nodes. Additionally, as mentioned in Section III-B, a Management Node controls VM consolidation, and a Power Monitor Node collects power consumption. These nodes are connected to a private network segment. The host nodes are also connected to a migration network segment, which is intended to isolate busty migration traffic from other management traffic. In our experiments, our consolidation system controls 5 VMs; each VM has one virtual CPU core and 1 GB RAM.

We developed a workload generator program running on a guest operating system. The packing system consolidates VMs in response to their CPU loads. Live migrations are deeply affected by their memory update speeds. To identify characteristics of our consolidation system, therefore, the program can generate any specified CPU loads and memory update intensities by interlacing short busy loops and sleeps. It is designed to emulate a server-type workload like web/mail applications. A small computational task is periodically generated at a calculated average rate conforming to the Poisson distribution; as is well known, the arrival rate of a new request to a network application is basically explained by the Poisson distribution.

B. Simple Scenario

First, we evaluate the basic effectiveness of using postcopy live migration for dynamic consolidation. In this evaluation, we used a simple load change scenario as shown in Figure 5. The load of VM0 is first set to 80%, and then reset to 40% at 75 seconds. The loads of other VMs are first set to 0.05%, and then the load of VM1 is reset to 30% at 175 seconds. The memory update intensity of workloads is set to 0.6; with this value, the memory update speed at a 100% CPU usage is approximately 200MB/s.

Figure 6 shows the CPU usage of host nodes and VMs. Figure 7 shows the power consumption of host nodes and network switches.

The left side of the figures shows the case of using postcopy live migration. At 85 seconds, the consolidation

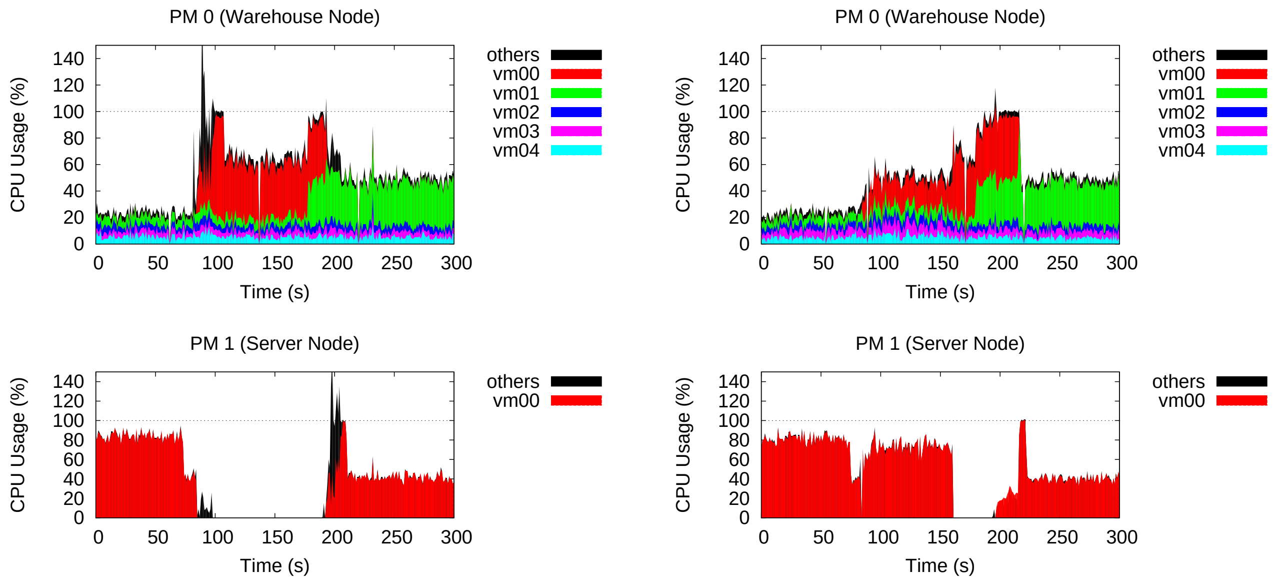

Figure 6. The CPU usage of Host Nodes and VMs (left: using postcopy, right: using precopy)

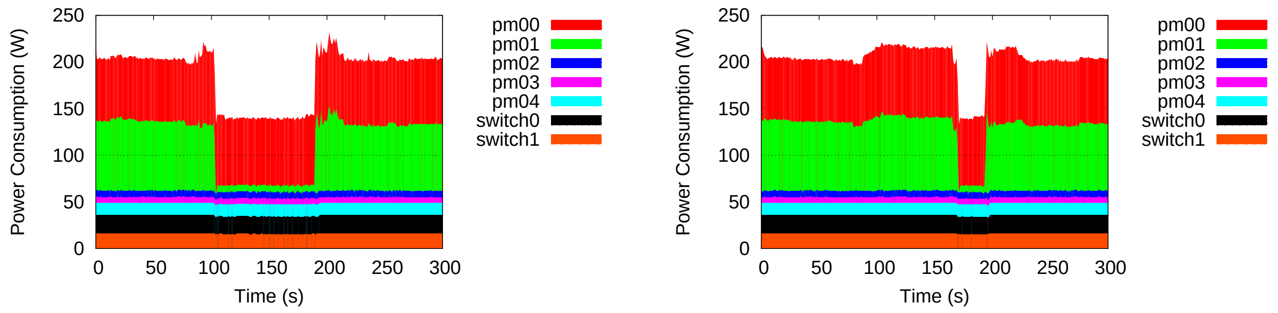

Figure 7. The Power Consumption of Host Nodes and Network Switches (left: using postcopy, right: using precopy)

system decided to consolidate all VMs into Warehouse Node (PM0), and then started to relocate VM0 to it. This live migration finished approximately at 100 seconds, and then Server Node (PM1) was suspended. As shown in Figure 7, the total power consumption was reduced by approximately 60W. It should be noted that the live migration incurred energy overheads (i.e., 20W or less) until completed; however, the overheads were far less than the power consumption saved by this dynamic consolidation. At 175 seconds, VM1 started consuming 40% CPU usage. After detecting the overloading of Warehouse Node (PM0), the consolidation system resumed Server Node (PM1) again, and relocated the most CPU-consuming VM (VM0) to it.

As shown in the right side of the figures, the consolidation system with precopy live migration started to relocate VM0 to Warehouse Node (PM0) at the same time as using postcopy (i.e., at 85 seconds). In this case, however, the live migration did not finish until 160 seconds. The memory update speed of VM0 was over 80 MB/s during the migration, which was relatively close to the available bandwidth (approximately 120 MB/s) of the migration network. Because precopy live migration needs to transfer updated memory pages repeatedly, the consolidation system could not promptly relocate VM0 to Warehouse Node (PM0) as performed with postcopy live migration. Resultingly, Server Node (PM1) was suspended only in 20 seconds (i.e., approximately 25% of using postcopy). In addition, energy overheads of the live migration were higher than using postcopy. A 20W increase of power consumption continued until the migration was completed, which was involved by dirty page tracking of the migrating VM.

Through these experiments, we confirmed that our consolidation system with postcopy live migration successfully worked, which dynamically optimized VM locations and server power states. In comparison with precopy live migration, postcopy live migration allowed the consolidation system to eliminate excessive power usage more aggressively for memory-intensive VMs.

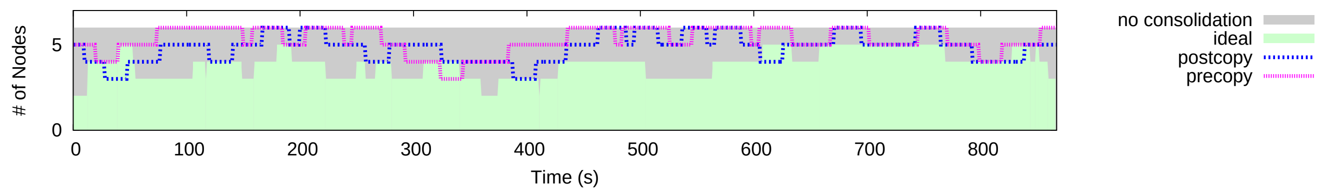

Figure 8. The Number of Active Host Nodes

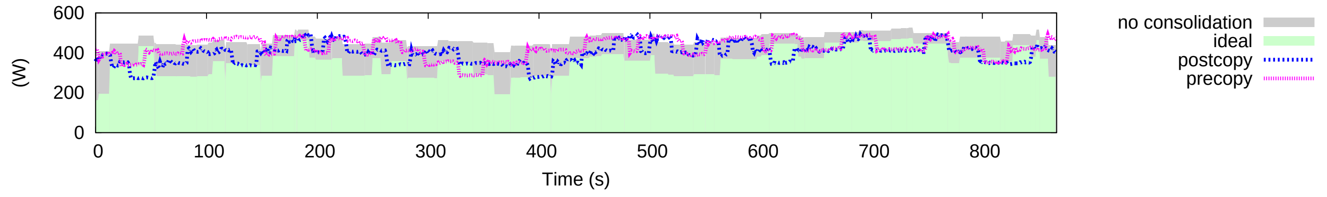

Figure 9. The Total Power Consumption of Host Nodes and Network Switches

C. Complex Scenario

Next, we evaluated our consolidation system with a compound load change scenario. We randomly generated an approximately 15-minutes scenario with the following rules, considering *race-to-halt*-like workloads. A workload on each VM changes its state between active and idle modes at 75% and 25% probabilities, respectively. A new mode continues for a random duration between 30 and 60 seconds. The workload generates a random CPU load between 70% and 100% in the active mode, and between 0% and 30% in the idle mode. The memory update intensity of workloads is set to 0.6, the same value as the previous experiments.

Figure 8 shows the number of active host nodes. **ideal** shows the theoretical number of host nodes required to pack all VMs at each time step, which is calculated from the load change scenario by using First Fit Algorithm; this number assumes that all migrations finish instantaneously at any time.

The consolidation system with postcopy live migration basically used fewer active host nodes than that of using precopy live migration. In the case of using precopy, a live migration sometimes prevented other following migrations from being started for a long time; VM locations were not sometimes optimized in response to load changes.

As shown in Figure 9, the consolidation system with postcopy live migration more closely fits to the ideal total power consumption, which is estimated on the assumption that the power consumption of a host node is proportional to the total CPU loads generated by VM workloads on it [1].

Table IV summarizes the total power consumption accumulated during the load change scenario. The power consumption was reduced by 11.8% with postcopy live migration, and by 5.2% with precopy live migration. It should be noted that the consolidation system addresses energy consumption overheads between the ideal case and no consolidation case; the consolidation system with postcopy live migration eliminated approximately half of the energy overheads, which is improved by approximately 50% from that of using precopy live migration.

[1]From Table III, the power consumption of a host node is roughly estimated to be at $53 + (100 - 53) * L$, where L is the total CPU loads generated by VM workloads on it.

Table IV
ACCUMULATED ENERGY CONSUMPTION

	Energy (Ws)	Saved Energy (%)
no consolidation	390175	-
ideal	294033	24.6
postcopy	344204	11.8
precopy	369877	5.2

V. Related Work

A. Postcopy Live Migration

SnowFlock [1] provides a VM cloning system enabling developers to easily program distributed systems. A postcopy technique is used to rapidly copy the state of a master VM to worker VMs. It is required to modify the memory management code of the Xen's hypervisor and the paravirtualized Linux system. A study [2] developed a postcopy live migration mechanism for the paravirtualization mode of Xen, which extends the swap-in/out code of the Linux kernel for on-demand memory transfer. A special device driver is required to be installed into the guest Linux system.

As described in our previous work [3], we have developed a postcopy live migration mechanism for KVM. In comparison with the above work, our mechanism supports guest operating systems without any modifications to them (i.e, no special device drivers and programs are needed in VMs); all guest operating systems including Windows, Linux, and *BSD are supported. It is implemented as a lightweight extension to KVM. It is not required to modify critical parts of the VMM code. We named our postcopy migration mechanism as Yabusame, and are now preparing to publish its source code under an open source license [15].

B. Dynamic VM Consolidation using Precopy Live Migration

To the best of our knowledge, this paper is the first work exploiting postcopy live migration for energy-efficient VM consolidation. The following studies regarding VM consolidation are based on precopy live migration.

Sandpiper [16] is a consolidation management system that dynamically optimizes VM locations in order to remove host overloading. This study showed that using workload-specific activity data, such as request arrival rates and response time, makes more optimized relocations possible; resource demand of VMs is estimated and predicted by queuing theory and autoregression analysis. In [17], a consolidation system uses a threshold value of resource usage to trigger VM repacking; if the CPU usage of a host exceeds this value, the system reoptimizes VM locations, so that mitigates the risk that application response times (e.g., service level agreement in this study) are adversely affected. The study [18] exploits an anomaly detection technique based a stochastic model, which determines the VMs and hosts subject to significant state changes. This study argues that a threshold-based algorithm incorrectly detects overloading and mistakenly determines a reconfiguration plan. The study [19] discusses the way of finding turning points of resource demands, where reconfiguration of VM locations is advisable. This technique aims to determine whether repacking is required or not with small calculation cost. Entropy [20] is a VM packing management system exploiting constraint programming techniques. It first determines the minimum number of nodes that are necessary to host all VMs, and then computes an optimal order of migrations to minimizing the overall reconfiguration time. The study [21] presents a network-aware migration scheduling algorithm, which tries to minimize the bandwidth usage while holding migration deadlines.

We consider that these techniques are also applicable to our consolidation system. We can extend our current packing algorithm with the above techniques, thereby improving scalability of our consolidation system for large-scale data centers. In future work, we will discuss the advantage of postcopy migration with other packing algorithms. In our previous work [22], we experimentally developed a genetic algorithm that determines near-optimal VM locations quickly. We have a plan to apply this algorithm to our consolidation system.

VI. CONCLUSION

In this paper, we have proposed an energy-efficient VM consolidation system exploiting postcopy live migration. Postcopy live migration greatly contributes to eliminating excessive power consumption, which allows our consolidation system to aggressively optimize VM locations. In postcopy live migration, the whole migration process finishes much more quickly than precopy live migration. We developed the prototype of our consolidation system, where excessive hardware nodes were suspended by means of ACPI S3 and all power usages were monitored with watt meters. Our experiments showed that our consolidation system with postcopy live migration eliminated more excessive power consumption than that of using precopy live migration. Postcopy live migration allowed the prototype system to eliminate 11.8 % energy overheads of actively-running VMs, which was improved by approximately 50% from precopy live migration.

In future work, we have a plan to integrate our consolidation mechanism into an open source cloud management system such as Eucalyptus [23] and OpenStack [24]. In addition, we are now preparing to apply our consolidation mechanism to a large-scale data center composed of hundreds of physical hosts. Further details will be reported in our upcoming papers.

ACKNOWLEDGMENT

This work was partially supported by KAKENHI (20700038 and 23700048) and JST/CREST ULP.

REFERENCES

[1] H. A. Lagar-Cavilla, J. A. Whitney, A. Scannell, P. Patchin, S. M. Rumble, E. de Lara, M. Brudno, and M. Satyanarayanan, "SnowFlock: Rapid Virtual Machine Cloning for Cloud Computing," in *Proceedings of the fourth ACM european conference on Computer systems*, Apr 2009, pp. 1–12.

[2] M. R. Hines and K. Gopalan, "Post-copy based live virtual machine migration using adaptive pre-paging and dynamic self-ballooning," in *Proceedings of the 5th International Conference on Virtual Execution Environments*, Mar 2009, pp. 51–60.

[3] T. Hirofuchi, H. Nakada, S. Itoh, and S. Sekiguchi, "Enabling instantaneous relocation of virtual machines with a lightweight VMM extension," in *Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, May 2010, pp. 73–83.

[4] A. Kivity, Y. Kamay, D. Laor, and A. Liguori, "kvm: the Linux virtual machine monitor," in *Proceedings of the Linux Symposium*, Jul 2007, pp. 225–230.

[5] T. Hirofuchi, H. Nakada, S. Itoh, and S. Sekiguchi, "Reactive consolidation of virtual machines enabled by postcopy live migration," in *Proceedings of the 5th International Workshop on Virtualization Technologies in Distributed Computing*, Jun 2011, pp. 11–18.

[6] Hewlett-Packard Corporation, Intel Corporation, Microsoft Corporation, Phoenix Technologies Ltd., and Toshiba Corporation, *Advanced Configuration and Power Interface Specification*, Apr. 2010.

[7] L. A. Barroso and U. Hölzle, “The case for energy-proportional computing,” *Computer*, vol. 40, pp. 33–37, Dec 2007.

[8] K. Cline, L. Grindstaff, S. Grobman, and Y. Rasheed, “Innovating above and beyond standards,” *Intel Technology Journal*, vol. 12, no. 04, pp. 255–268, 2008.

[9] Intel Corporation, Hewlett-Packard Company, NEC Corporation, and Dell Computer Corporation, *-IPMI- Intelligent Platform Management Interface Specification Second Generation v2.0*, Feb. 2004.

[10] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, “Xen and the art of virtualization,” in *Proceedings of the nineteenth ACM symposium on Operating systems principles*, 2003, pp. 164–177.

[11] J. Sugerman, G. Venkitachalam, and B.-H. Lim, “Virtualizing I/O devices on VMware workstation’s hosted virtual machine monitor,” in *Proceedings of USENIX Annual Technical Conference*, 2001, pp. 1–14.

[12] M. Nelson, B.-H. Lim, and G. Hutchins, “Fast transparent migration for virtual machines,” in *Proceedings of USENIX Annual Technical Conference*, 2005, pp. 25–25.

[13] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, “Live migration of virtual machines,” in *Proceedings of the 2nd Symposium on Networked Systems Design and Implementation*, 2005, pp. 273–286.

[14] A. Mirkin, A. Kuznetsov, and K. Kolyshkin, “Containers checkpointing and live migration,” in *Proceedings of the Linux Symposium 2008*, Jul 2008, pp. 85–92.

[15] AIST Cloud Computing Research, http://grivon.apgrid.org/; Last accessed: June 30, 2011.

[16] T. Wood, P. J. Shenoy, A. Venkataramani, and M. S. Yousif, “Black-box and gray-box strategies for virtual machine migration,” in *Proceedings of the 4th Symposium on Networked Systems Design and Implementation*, 2007, pp. 229–242.

[17] G. Khanna, K. Beaty, G. Kar, and A. Kochut, “Application performance management in virtualized server environments,” in *Proceedings of the IEEE/IFIP Network Operations and Management Symposium*, Apr 2006, pp. 373–381.

[18] M. Andreolini, S. Casolari, M. Colajanni, and M. Messori, “Dynamic load management of virtual machines in a cloud architectures,” in *Proceedings of the IEEE Conference on Cloud Computing*, Oct 2009.

[19] T. Setzer and A. Stage, “Decision support for virtual machine reassignments in enterprise data centers,” in *Proceedings of the 5th IEEE/IFIP International Workshop on Business-driven IT Management*, Apr 2010.

[20] F. Hermenier, X. Lorca, J.-M. Menaud, G. Muller, and J. L. Lawall, “Entropy: a consolidation manager for clusters,” in *Proceedings of the 5th International Conference on Virtual Execution Environments*, 2009, pp. 41–50.

[21] A. Stage and T. Setzer, “Network-aware migration control and scheduling of differentiated virtual machine workloads,” in *Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing*, 2009, pp. 9–14.

[22] H. Nakada, T. Hirofuchi, H. Ogawa, and S. Itoh, “Toward virtual machine packing optimization based on genetic algorithm,” in *Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living*, ser. Lecture Notes in Computer Science, vol. 5518, Jun 2009, pp. 651–654.

[23] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, “The eucalyptus open-source cloud-computing system,” in *Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, 2009, pp. 124–131.

[24] The OpenStack Project, http://www.openstack.org/; Last accessed: June 30, 2011.