IPSJ Transactions on Advanced Computing Systems Vol. 3 No. 2 138–146 (June 2010)

Regular Paper

A Rank-based VM Consolidation Method for Power Saving in Datacenters

Shingo Takeda[†1] and Toshinori Takemura[†2]

In this paper, we propose a simple but flexible virtual machine consolidation method for power saving. This method is specifically designed for datacenters where heterogeneous high-density blade servers host dozens or even hundreds of virtual machines. This method utilizes an extended First-Fit Decreasing (FFD) algorithm. It selects a migration destination server on the basis of server rank. The rank represents server selection priority and is uniquely assigned to each physical server. Our simulation results show that this method reduces power consumption by 34.5% under a typical workload and 33.8% under a random workload.

1. Introduction

Current trends toward centralizing business IT infrastructures have made power-saving technology a critical need for datacenters. Datacenters hosting user desktops and software for service (SaaS) applications are using increasing numbers of servers. These datacenters consume large amounts of electricity not only for operating the IT devices but also for cooling them [1].

Virtualization is an important technology for saving power. It allows running multiple virtual machine (VM) instances on a single physical machine (PM). A datacenter operator is able to reduce PMs by consolidating low-load VMs onto a fewer number of PMs. Virtual machine monitors (VMMs) such as Xen hypervisor (Xen) [2], Kernel-based Virtual Machine (KVM) [3], VMware ESX [4] and Hyper-V [5] supports live migration of running VMs across different PMs over a network. The downtime from live migration may be as a few dozen milliseconds which is not noticeable by the application user. Live migration enables dynamic VM remapping in response to fluctuating workloads.

High-density blade servers are ideal for hosting dozens or even hundreds of VMs. A blade server is designed to save energy and installation space by sharing power supply units and cooling fans with several compact servers. Power effectiveness of blade servers has been improved by advanced semiconductor process and processor architecture. When new servers are added to a datacenter for capacity extension, the power effectiveness of new and existing servers can be significantly different. A technology that manages heterogeneous servers and optimizes total energy consumption of a datacenter is therefore greatly needed.

In this paper, we propose an autonomic VM consolidation method for power saving in datacenters. The method enables flexible VM remapping to optimize power consumption. The rest of the paper consists as follows. We examine how blade servers consume power in Section 2. We examine CPU and network load caused by live migration in Section 3. We describe our consolidation method in Section 4 and evaluate it with a simulator in Section 5. We explain the difference between related works and ours in Section 6. Finally, we conclude the paper in Section 7.

2. Power Consumption of Blade Servers

Blade servers save energy and space by sharing power supply units (PSUs) and cooling fans. The layout of a blade server modeled Express5800/120Ba-4 (single Xeon 3.6 GHz, 8 GB memory, one HDD and two 1000BASE-T ports) is shown in **Fig. 1**. The blade unit consists of 6 blade servers, 2 Ethernet switches, 3 redundant PSUs, 5 cooling fans (3 of them are behind it so are not shown here) and a blade enclosure.

We measured power consumption of the blade unit. The sum of power consumption observed at the three PSUs is shown in **Fig. 2**. It consumes at least 343.9 watts for PSUs, Ethernet switches, cooling fans and an enclosure. We refer to this power as $P_{Enclosure}$. Each server consumes 162.2 watts during maximum CPU usage.

To examine accurate power consumption of a server, we also measured power consumption of a stand-alone server modeled Express5800/110El (dual-core Xeon E3110, 8 GB memory and 3 HDDs). The power consumption of the server running

†1 1st Computers Software Division, NEC Corporation
†2 Service Platforms Research Laboratories, NEC Corporation

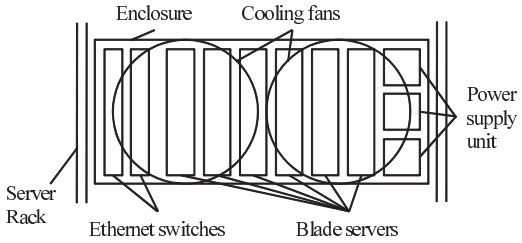

Fig. 1 Blade server layout.

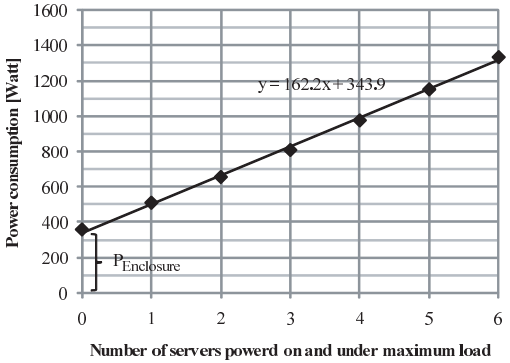

Fig. 2 Power consumption of blade unit.

one or two threads of workload with CPU usage limit is shown in **Fig. 3**. The server consumes at least 135.8 watts. We refer to this base power as P_{Base}. Power consumption of the first core varies linearly up to 22.1 watts depending on CPU usage. However, power consumption of the second core varies less. We refer to this varying power depending on CPU usage $u_{CPU}(t)$ as $p(u_{CPU}(t))$. As effects of other factors, such as disk access rate and fan speed, were relatively small, we assume that they can be included statically in P_{Base}.

If $P_{Enclosure}$, P_{Base} and $p(u_{CPU}(t))$ are given, we can roughly estimate the total power consumption of a blade unit P_{Unit} with the equation below.

$$P_{Unit} = P_{Enclosure} + \sum_{All\ active\ servers} (P_{Base} + p(u_{CPU}(t)))$$

The basic idea behind power saving is to cut off P_{Base} by emptying servers

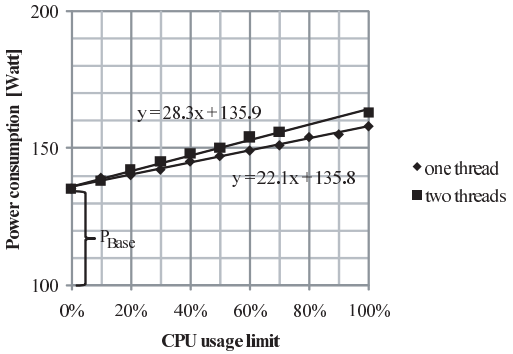

Fig. 3 Power consumption of stand-alone server.

through live migration so they can be shut down. Also, more power can be saved by selecting servers having smaller P_{Base} and better value of function p as migration destinations in heterogeneous environment. Furthermore, cooling efficiency can be improved by physically separating the heat sources.

3. Overhead of Live Migration

Live migration stresses the CPU and network at both the source and destination during transfer of VM memory images. In autonomic remapping, these stresses must be counted so as not to degrade total system performance due to excessive migration.

The Xen live migration method is called iterative pre-copy [6]. Before suspending a VM for server switching, Xen transfers a memory image from a source server to a destination server to minimize downtime. Updated (dirtied) memory pages are re-transferred iteratively during the transfer until the remaining dirty pages are a small enough quantity to allow prompt (few dozen milliseconds) switching. Therefore, more data is transferred over the network than in the actual memory image. The live migration methods of KVM and Hyper-V are basically the same [5),7)].

We examined CPU and network usage by the pre-copy method with Xen 3.3.0. We monitored the resource usages on the blade servers described in Section 2 with the simple tool that we developed with the libxenstat library included in

the Xen package. The servers were connected through 1000BASE-T via an Ethernet switch module installed in the enclosure. The disk image is not transferred between servers because it is shared on an NFS server.

The configuration of migration test is shown in **Fig. 4**. Each VM has 512 MB memory allocated. We moved VM-M from the blade server 1 to 2 monitoring Xen on the blade server 1. We tested 8 ($= 2^3$) combinations of VM load. Each of the 3 VMs were in either one of two states which are idle (0% CPU) or busy (100% CPU; kernel compiling). Each combination was tested 3 times (total of 24 times).

The result is shown in **Table 1**. Wallclock time depended on server load. It became longer when any VM's CPU usage became higher. However, CPU time was independent from load of any VM. This means that 512 MB migration always uses the CPU for 3 seconds which appears as a CPU usage of 5% during 60 second monitoring intervals. Network usage was independent from load of VM-A and VM-B, but was dependent on load of VM-M. The reason is that a high dirtying rate causes much re-transfer. Assuming the dirtying rate is proportional to the target VM CPU usage u_{VM} and is not changing, we can estimate the network usage $U_{Network}$ during 60 second monitoring intervals. The equation below is an example.

$$U_{Network} = 8 \times (544 + (657 - 544) \times u_{VM})/(60 \times 1{,}000)$$

The network usage increases 8% if the VM CPU usage is 50%.

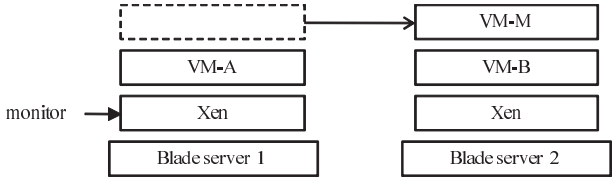

Fig. 4 Migration test configuration.

Table 1 Overhead for live-migrating a 512 MB-memory VM with 1,000 Mbps LAN.

	min	avg	max	correlation
Wallclock time [s]	8.0	16.3	22.4	depends on CPU usage of every VM
CPU time [s]	2.7	3.0	3.3	independent from CPU usage of any VM
Network [MB]	544	561	657	depends on CPU usage of VM-M

Though predicting wallclock time is not easy, we can estimate the CPU and memory usage from these results if the memory size and CPU usage of the target are known.

4. Proposed Consolidation Method

We propose an autonomic remapping method for power saving in datacenters using dozens or even hundreds of VMs that provide various applications and services. The architectural design of the method is shown in **Fig. 5**. It is centralized because we place importance on manageability. An agent is deployed on every PM and a manager on the dedicated management PM. The agent periodically monitors resource capacities and usages of PM (1), and reports it to the manager at intervals (e.g., 30 seconds) (2). The manager periodically remaps VMs in response to the reported capacities and usages at intervals (e.g., 1 minute) (3). The manager sends live migration orders to PMs via agents (4, 5) as needed. The manager also changes the PM power state (6).

4.1 Server Ranks

Every PM has a server rank which is a unique value representing selection priority of the PM. An example of the ranks designed for minimizing power

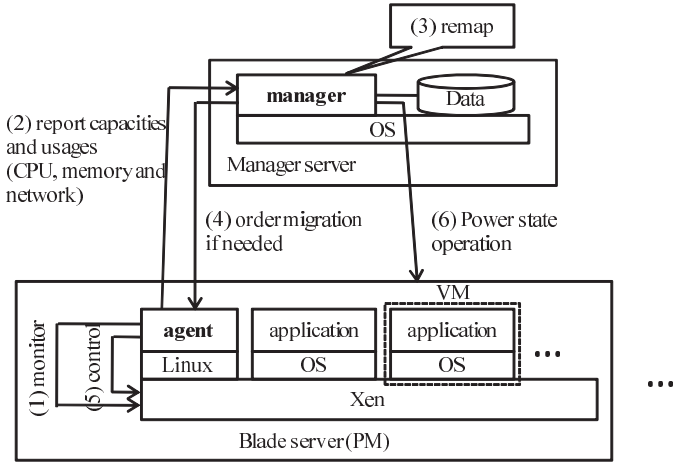

Fig. 5 Architectural design of proposed method.

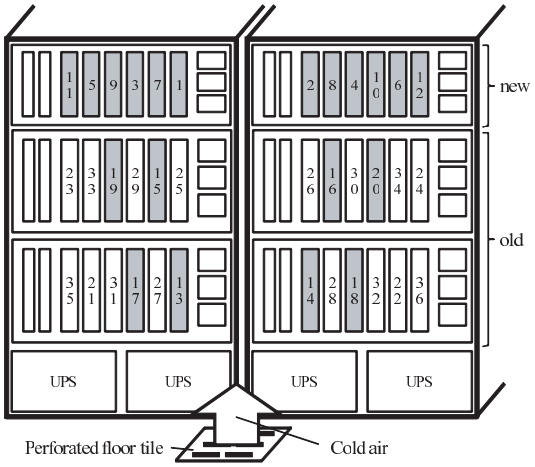

Fig. 6 Server ranks assigned to minimize power consumption. The top 20 servers are highlighted.

consumption is shown in **Fig. 6**. The lower 4 blade units are an old model whose power effectiveness is worse than the upper two new blade units. Higher priorities are assigned to the new servers to make use of them. We also use the ranks to avoid making hotspots. The example shows that the ranks start from the side near the cold air. Also the ranks are set to preferably avoid using servers physically located next to each other. These improve cooling efficiency and server stability. Basically the ranks are decided by datacenter operator because there are a lot of physical factors inaccessible to the software.

The server ranks can be dynamically changed during operation. For example, an operator gives low priority to a PM whose reliability is considered low. Another example is that an operator rotates ranks in order to prevent frequent power state switch from damaging a particular PM.

4.2 Resource Allocation

In advance to remapping, resource allocations for each VM need to be decided. We consider the following three classes of PM resource capacity as constraints.

- CPU performance C_{CPU} (e.g., 3,600 MHz)
- Memory size C_{Memory} (e.g., 8 GB)
- Network throughput $C_{Network}$ (e.g., 1,000 Mbps)

We assume that storage is shared among all PMs and do not consider disk I/O bandwidth as a constraint.

Our method uses the 95th percentile of recent usage to decide resource allocation. As the 95th percentile closely reflects the needed capacity, many Internet service providers (ISPs) use it for billing. It adapts well to a workload fluctuating at a high frequency. It is calculated with a light-weight algorithm without long-term historical data. For these reasons, the manager is able to decide resource allocations quickly and remap the VM at short intervals.

4.3 VM Remapping

Mapping VMs on the minimum number of PMs with resource constraints is one of bin-packing problems. Strictly deriving the optimal solution of a bin-packing problem is known as NP-hard. As tens or hundreds of VMs run in a datacenter, it is impractical for the manager to find the optimal layout within an interval.

Our remapping algorithm is similar to First-Fit Decreasing (FFD) algorithm which is a widely used to solve bin-packing problems. Although FFD does not derive the optimal solution, it is fast and efficient. It was recently proven that the number of bins (PMs) given by FFD is no more than $11/9OPT + 6/9$ where OPT is the number of bins given by the optimal solution [8]. As examined in Section 3, migration stresses the CPU and network. A remapping algorithm is required to decide effective pairs of migration source and destination PM in order not to squander the two resources.

The simplified illustrations of our remapping algorithm are shown in **Fig. 7**. Our algorithm packs a large object (highly loaded VM) into the first bin (the PM with highest rank) in which it will fit, the same as the FFD algorithm does. However in order to avoid ineffective migrations this algorithm does not totally reset the current mapping. We introduce two thresholds R_{High} and R_{Low}. We refer to a PM at least one of whose resource usages is over R_{High} as a high-load PM. Conversely, we refer to a PM all of whose resource usages are below R_{Low} as a low-load PM. In our algorithm, only VMs hosted by high-load or low-load PM become migration candidates. In Fig. 7, only VM3 is a candidate in the left case, and VM3 and VM4 are candidates in the right case. The figure shows how the migration destinations for VM3 are decided. The algorithm does both load

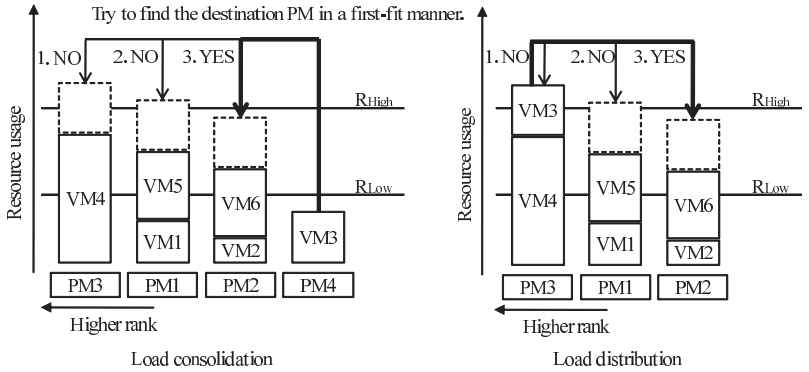

Fig. 7 Simplified illustrations of remapping algorithm. Only one resource is shown.

```
for each interval {
  create pm_list in ascending order of rank
  create vm_list in descending order of resource usage

  for each vm in vm_list {
    host := get current host of vm
    if (host is high-load) or (host is low-load) {
      for each pm in pm_list {
        if (pm is not sending a VM memory for migration)
            and (pm is not receiving a VM memory for migration)
            and (pm is capable to host vm) {
          start moving vm from host to pm
        }
      }
    }
  }
}
```

Fig. 8 Pseudo-code of remapping algorithm.

consolidation and distribution in the same fashion.

The pseudo-code of our remapping algorithm is shown in **Fig. 8**. The term capable in the figure means that the destination candidate PM will not be high-load after migration. In order to maintain efficiency, multiple VM are not simultaneously allowed to migrate to a PM.

```
for each interval {
  create pm_list in ascending order of rank

  n_empty := count empty pm in pm_list
  n := n_empty - n_standby

  for each pm in pm_list {
    if (pm is powered off) and (n < 0) {
      boot pm
      n := n + 1
    }
  }
  for each pm in reversed pm_list {
    if (pm is powered on) and (n > 0)
       and (pm has been empty for more than 10 mins) {
      shutdown pm
      n := n - 1
    }
  }
}
```

Fig. 9 Pseudo-code of power state operation algorithm.

4.4 Power State Operation

Our method shuts down empty PMs to cut off P_{Base} if possible. Here, an empty PM is a PM which is not hosting any VM. The pseudo-code of our power operation algorithm is shown in **Fig. 9**. The manager keeps at least $N_{Standby}$ empty PMs powered on to prepare for workload rise. These stand-by PMs are selected in order of rank. The manager waits for a while after a PM becomes empty, because excessive power operation might damage the hardware.

5. Evaluation

We evaluated our method with a simulator which we implemented in Java. The simulator reads workload data of VMs and remaps the VMs by the method described in Section 4.

5.1 Simulation Configuration

The server configuration and ranks are the same as Fig. 6. The ranks were fixed to maximize consolidation effectiveness. Every PM has a 3,600 MHz single-

core CPU, 8,096 MB memory and a 1,000 Mbps network interface. The VMM exclusively uses 256 MB of the memory. The new PM consumes 120 watts maximum and 60 watts minimum. The old PM consumes 200 watts maximum and 100 watts minimum. The power consumption varies linearly depending on CPU usage. Each of the new enclosures consumes a constant 200 watts, and the old one does 350 watts.

The manager processes 100 VMs in one minute intervals. The manager uses the 95th percentile calculated from resource usages of past 10 minutes by linear interpolation. The thresholds R_{High} and R_{Low} are fixed at 70% and 40% respectively at this time because they are good values according to our experience. Initially, the PMs are all up and the VMs are mapped in advance with FFD algorithm. Every PM requires 3 minutes for booting and 2 minutes for shutdown. In addition, every PM consumes maximum power while booting and shutdown. The number of stand-by PM ($N_{Standby}$) is 1.

5.2 Workload Input

We ran the simulator with two sets of workload data, typical workload data and random workload data. Both data consist of 30-second-interval timestamp, CPU cycle (MHz), memory amount (MB) and network traffic (Mbps) for 100 VMs for a 24 hour period.

The typical workload data was created artificially. We created 9 models of data by examining the workloads of actual enterprise servers such as WWW, batch, retail, stock market and streaming server. Each model is used by 11 or 12 VMs. An example of WWW workload data is shown in **Fig. 10**. The workload has two peaks in the morning and evening. The total workload of 100 VMs has a large peak at 8:00.

The random workload data was randomly generated by Markov Modulated Poisson Process (MMPP). The MMPP is one method for generating bursty traffic observed in many Internet services [9]. A transition table and an arrival rate table are also generated randomly for each of 100 VMs. An example of the workload data is shown in Fig. 10. Since fluctuations are unpredictable, remapping is considered more difficult than the previous case.

5.3 Overhead Emulation

The simulator calculates CPU and network overhead of migration based on the

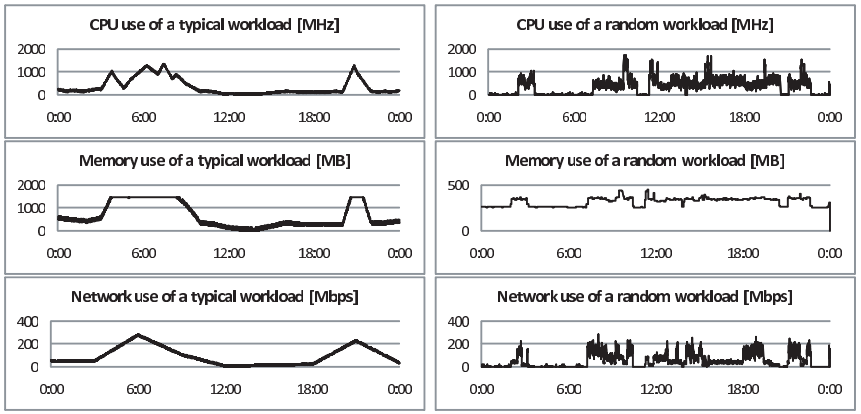

Fig. 10 Examples of typical and random workload data. The example typical data is one of 9. The example random data is one of 100.

results in Section 3. We assume that every migration finishes within the remapping interval because each VM memory size is always smaller than 1,024 MB. We also assumed that CPU time used for a migration is proportional to VM memory size. A VM with 512 MB memory uses the CPU for 3 seconds. The network load is proportional to VM memory size and increases up to 25% depending on CPU usage. The migration overheads of the source and destination are symmetric.

5.4 Results

The manager consolidated VMs on PMs having higher ranks under both typical and random workloads. A screenshot of the simulator is shown in **Fig. 11**. There are 36 numbered rectangles each representing a PM. The server resource usages are shown as a bar graph in each rectangle. The left bar represents CPU usage, the middle is memory usage and the right is network usage. The two horizontal lines drawn in the rectangle are the thresholds R_{High} and R_{Low}. The rank-27 PM was booting and under maximum CPU usage at the time because there was no stand-by PM. In addition, every simulation finished within 10 seconds with a Pentium 4 PC.

The system resource usages under typical workload and the simulated power consumption are shown in **Fig. 12**. The manager on average used 34.5less system power compared to the assumed power usage when using the PMs equally without

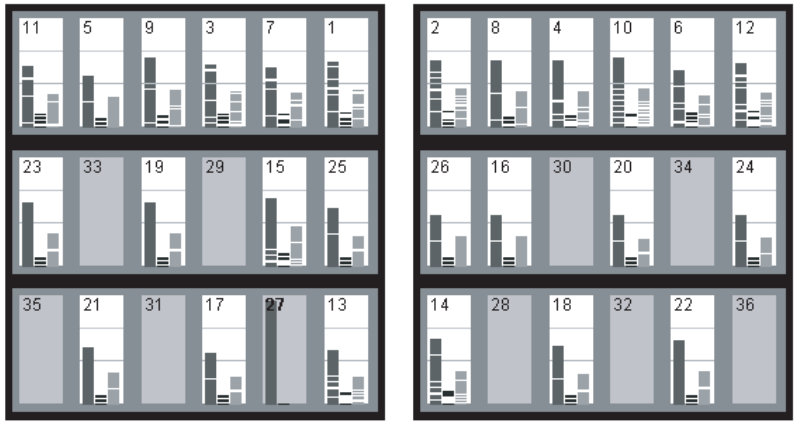

Fig. 11 Screenshot of simulator with typical workload.

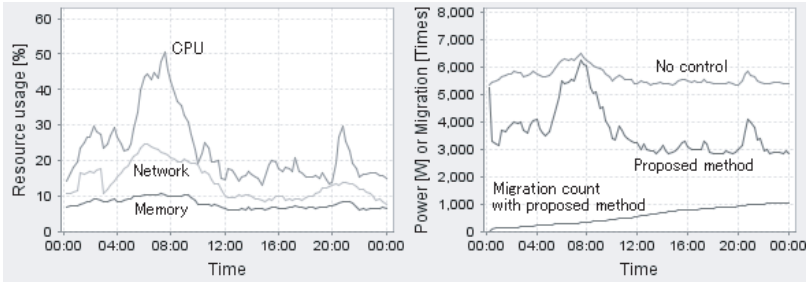

Fig. 12 Total resource usages under typical workloads and power consumption of 36 PMs.

migration. The manager ordered migration 1,040 times, power-on 68 times and power-off 92 times. A VM was hosted by over-load PMs which resource usage is over 80% for 0.01% of 24 hour period on average. We refer to this rate as the over-load rate. The power saving rate dropped to 22.59% when we reversed the ranks to minimize utilization of power-effective PMs.

The system resource usage of random workload and simulated power consumption are shown in **Fig. 13**. The manager on average used 33.78% less system power by ordering migration 2,183 times, power-on 55 times and power-off 73 times. The over-load rate was 0.31%. The power saving rate dropped to 20.86%

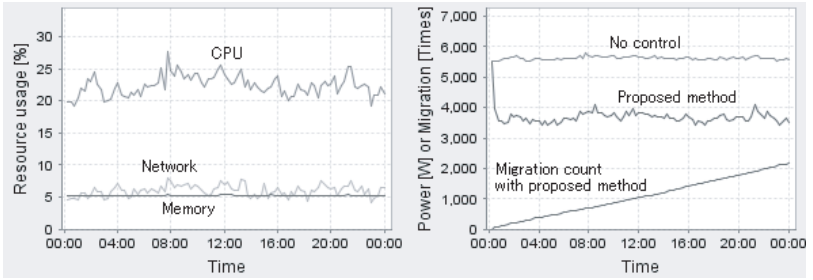

Fig. 13 Total resource usages under random workloads and power consumption of 36 PMs.

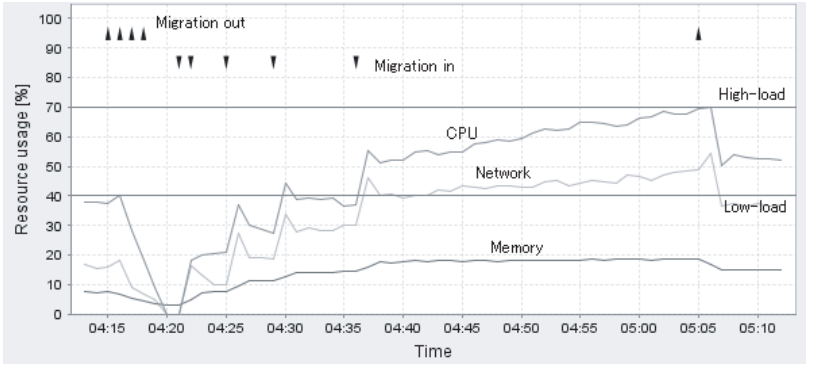

Fig. 14 Resource usages and migration points over time of a PM. Range of the X-axis is 1 hour.

when we reversed the ranks.

The resource usage and migration points of a PM over time are shown in detail in **Fig. 14**. An upward triangle indicates that the manager moved a VM to another PM, and a downward triangle indicates that the manager moved a VM from another PM. The manager first moved 4 VMs out to empty the PM because it became low-load. After 04:20, the manager decided to use the PM again and moved 5 VMs in. At 05:05, the manager moved 1 VM out because the PM became high-load. The spikes on CPU and network usage are migration overhead.

5.5 Discussion and Future Work

The result shows that the method is effective and practical. As Fig. 11 shows, the manager consolidates VMs on high-rank PMs, shuts down empty PMs and scatters the heat sources. This reduces the system power consumption and improves cooling effectiveness. The system consumes a great deal of power ($\approx$ 5,500 watts) in the first parts of the typical and random cases. After the manager controls the system, its power consumption is reduced. Figure 12 shows that power consumption ideally fluctuates in proportion to workload.

The result also shows that utilizing the ranks effectively save power in a heterogeneous environment. The power saving rates substantially decreased when the ranks were reversed in order to give high priority to old servers,

Our method is fast and does not require a high performance server for the manager. The result shows that 24-hour simulation finishes within 10 seconds with a PC. A manager server does not require special high-performance hardware.

Furthermore, the method does not significantly degrade system performance. The over-load rates for typical and random case were respectively 0.01% and 0.31%. While the fluctuation in the random workload is completely unpredictable, the manager adapts well by using the 95th percentile. The manager also uses R_{High} and R_{Low} to avoid excessive migration.

However, the numbers of migrations are large and should be decreased. Since there are 100 VMs, a VM is moved an average of once an hour during a random workload. While the migration overhead is acceptable (CPU 5% and network 8% for a minute) in our environment, the number of migrations can be a potential defect in environment with more overhead. We are considering changing R_{High} and R_{Low} dynamically using feedback from a migration counter.

6. Related Works

Liu, et al. proposed the GreenCloud architecture for saving datacenter power [10]. It uses a heuristic algorithm for VM consolidation. The algorithm minimizes the cost that is calculated from the number of migrations, number of online hosts and server utilization. The evaluation formula must be defined by an operator who has deep knowledge of his system and the algorithm. According to their evaluation, the algorithm can obtain a near-optimal solution for 5 VMs on 5 PMs in less than 300 milliseconds.

Nathuji, et al. implemented the VirtualPower Management (VPM) that consolidates VMs considering processor power effectiveness [11]. VPM uses modified Xen and provides application-specific and fine-grained power management with virtual hardware power states.

Hu, et al. proposed the Magnet that is a scheduling policy for power reduction in a heterogeneous scientific cluster [12]. It consolidates VMs on multilayer ring-based overlay network. The network is scalable but the topology is more complex than centralized architecture.

VMware Distributed Power Management (DPM) [13] is a commercial product and consolidates workloads onto fewer servers and powers off the rest of the servers to reduce power consumption. It is a software component of VMware vSphere. SigmaSystemCenter [14] also is a commercial product and has a similar function. The architectures of these products are centralized. The manager programs are designed to minimize the number of active PMs.

Our application-independent method uses the operator-friendly server rank for controlling tens or hundreds of VMs. Our method simply and efficiently consolidates VMs on high-rank PMs rather than making an effort to minimize the number of active PMs. Our simulator can remap 100 VMs on 36 PMs 1,440 times in less than 10 seconds. The rank allows datacenter operators to flexibly control their system. They can for example easily disperse the heat sources in order to avoid making hotspots and give low priority to relatively unreliable PMs at the same time.

7. Conclusion

We propose an autonomic VM consolidation method for power saving in datacenters. The method moves VMs on the basis of server ranks which are uniquely assigned to PMs. The method uses the 95th percentile and two thresholds not to degrade system performance. Our simulation result shows that the method is effective and practical. We anticipate that this method will be implemented in VM management software and will help datacenter operators save system power.

Acknowledgments We would like to thank the reviewers who gave us valuable comments and suggestions.

References

1) The green grid: *Guidelines for energy-efficient datacenters* (2007).
2) Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I. and Warfield, A.: Xen and the art of virtualization, *SOSP'03: Proc. 19th ACM symposium on Operating systems principles*, pp.164–177 (2003).
3) Qumranet Inc.: *KVM—Kernel-based Virtualization Machine* (2006).
4) VMware, Inc.: VMware ESXi 3.5 product datasheet. Available online at http://www.vmware.com/products/esxi/ as of September 2009.
5) Microsoft Corporation: *Windows Server 2008 R2 & Microsoft Hyper-V Server 2008 R2—Hyper-V Live Migration Overview & Architecture* (2009).
6) Clark, C., Fraser, K., Hand, S., Hansen, J.G., Jul, E., Limpach, C., Pratt, I. and Warfield, A.: Live migration of virtual machines, *NSDI'05: Proc. 2nd conference on Symposium on Networked Systems Design & Implementation*, pp.273–286 (2005).
7) KVM project: Migration—KVM. Available online at http://www.linux-kvm.org/page/Migration as of September 2009.
8) Dósa, G.: The Tight Bound of First Fit Decreasing Bin-Packing Algorithm Is $FFD(I) \leq (11/9)OPT(I) + 6/9$, *ESCAPE 2007, Springer LNCS 4614*, pp.1–11 (2007).
9) Hryn, G., Chydzinski, A. and Jerzak, Z.: MMPP-based HTTP traffic generation with multiple emulated sources (2004). Available online at http://wwwse.inf.tu-dresden.de/zib/publ/doc/hryn2004mmpp.pdf as of September 2009.
10) Liu, L., Wang, H., Liu, X., Jin, X., He, W.B., Wang, Q.B. and Chen, Y.: GreenCloud: a new architecture for green data center, *ICAC-INDST '09: Proc. 6th international conference industry session on Autonomic computing and communications industry session*, pp.29–38 (2009).
11) Nathuji, R. and Schwan, K.: VirtualPower: coordinated power management in virtualized enterprise systems, *SOSP '07: Proc. 21st ACM SIGOPS symposium on Operating systems principles*, pp.265–278 (2007).
12) Hu, L., Jin, H., Liao, X., Xiong, X. and Liu, H.: Magnet: A novel scheduling policy for power reduction in cluster with virtual machines, *2008 IEEE International Conference on Cluster Computing*, pp.13–22 (2008).
13) VMware, Inc.: *VMware Distributed Resource Scheduler* (*DRS*) (2009).
14) NEC Corporation: SigmaSystemCenter 2.1 Configuration Guide—Forth Edition (Update 2). Available online at http://www.nec.co.jp/pfsoft/sigmasystemcenter/doc/ConfigurationGuide_21E-4.pdf as of September 2009.

(Received September 25, 2009)
(Accepted January 15, 2010)

Shingo Takeda was born in 1980. He received his Ph.D. degree from Osaka University in 2008. Since 2008, he has been working for NEC Corporation. His interests include management techniques for cloud computing infrastructures. He is a member of IPSJ.

Toshinori Takemura was born in 1969. He received his M.E. degree from Nagoya University in 1994. Since 1994, he has been working for NEC Corporation. His research interests include performance evaluation and system management of cloud computing infrastructures. He is a member of IPSJ.