Computing Resource Transformation, Consolidation and Decomposition in Hybrid Clouds

Jinho Hwang
IBM T.J. Watson Research Center

ABSTRACT

With the promise of providing flexible and elastic computing resources on demand, the cloud computing has been attracting enterprises and individuals to migrate workloads in the legacy environment to the public/private/hybrid clouds. Also, cloud customers want to migrate between cloud providers with different requirements such as cost, performance, and manageability. However the workload migration is often interpreted as an image migration or re-installation/data copying as the exact snapshot of the source machine. Also the various cloud platforms and service models are rarely taken into consideration during the migration analytics. Therefore, although the expectation has risen with various requirements on the target cloud platforms and environments, the cloud migration techniques have not provided enough options that can satisfy the various requirements.

In this paper we propose a model to tackle the migration challenges that transform one resource into same or another resource in hybrid clouds. We formulate the problem as a constraint satisfaction problem, and iteratively decompose the server components and consolidate the servers. The ultimate goal is to recommend the optimal target cloud platform and environment with the minimum cost. Through the evaluation of the proposed model using the real enterprise dataset (up to 2012 machines), we prove that the proposed model satisfies the goal. We show that when migrating into virtualized cloud environments, the thorough resource planning can reduce 16% of current resources, about 5% - 10% servers can be consolidated, and more than 60% servers are possible candidates for server decomposition.

I. INTRODUCTION

Enterprises are increasingly migrating their existing IT infrastructure to the cloud, driven by the promise of low-cost access to elastic resources [1, 2]. Depending on the compliance and security requirements, as well as the business criticality of the business applications (often referred to as workloads), enterprises may choose one or more cloud environments, such as private cloud, 3rd party public cloud, traditional data center, resulting in hybrid cloud environments. Because the target environments are diversified and enterprises have a large number of servers, the migration planning quickly becomes a intractable problem. Also, as the competition becomes stronger, cloud providers increasingly offer more diversified services and they differentiate their catalogs with more advanced service features [3, 4].

A deterrent for enterprise migration to the cloud is a lack of migration planning tools that can scrutinize the discovered on-premise data, provide comprehensive analytical information to reason about why the migration can help reduce operational expenses and increase performance, and finally create a detailed migration plan [5, 6, 7]. Existing tools aim to only provide an one-to-one migration that just copies a source image into a target image, but they do not find themselves as the comprehensive end-to-end migration toolings [8, 9].

Technical challenges in migration are driven by the heterogeneity of the source and target environments, number of available tools and many unanticipated events and exceptions that can derail the process [10, 11, 12]. For example, availability of a large variety of server platforms impacts the choice of migration techniques that need to be considered. On-premise servers typically run on different platforms, different physical hardwares, and various hypervisors that involve different image formats. As a result, there is no one-size fits all migration approach, and this spurs the variety of methods [13, 14]. Challenges in managing end-to-end migration activities span from source discovery, resource planning, migration context sharing (e.g. target design, decisions, what/if analysis), exception handling, and timely notifications to process tracking. It is the most of importance that many of these processes can be smoothly executed with well planned resource mappings between the source and the target upfront.

In this paper, we propose a model to tackle the migration challenges that transform one resource into same or another resource in hybrid clouds based on source requirements and target availability. We investigate the tranformation metrics that need to be taken into consideration for the migration resource planning, server consolidation, and server decomposition. We formulate the problem as a constraint satisfaction problem (CSP), and iteratively decompose the server components and consolidate the servers. The ultimate goal is to recommend the optimal target cloud platform and environment with the minimum cost. The contributions in this paper include:

- identifying important metrics to decide target resources, server consolidation, and server decomposition;
- formulating to find the resource mappings with constratints;
- recommending servers that can be consolidated into the same machine or decomposed into multiple servers;
- evaluating the proposed methods with the real enterprise data.

II. ENTERPRISE-SCALE MIGRATION

Cloud transformation for an enterprise IT environment entails the processes that migrate an enterprise's data, applications, and services from on-premise data centers to a cloud environment, or to multiple cloud environments respectively

978-3-901882-77-7 © 2015 IFIP

managed by different cloud providers. A typical migration process starts with the discovery stage, followed by the analysis of the uncovered source environment and evaluation of its fitness to the target [5, 15]. For example, can the discovered resources be moved to the target cloud as they are or do they require some level of adaptation? The planning stage groups similar servers or workloads into so-called "waves" that will be scheduled for migration. Before migration is executed, pre-configuration is performed, which includes provisioning of target environment and network setup [16]. Once the migration is completed, post-configuration tasks are executed, such as backup switching. The process completes with the quality assurance step to ensure that the new environment is operational as expected [17].

At each step of migration processes, practitioners have a choice of one or more tools. Image based migration, is a per-unit based method, such as physical to virtual conversion [18]. Example of existing tools for this migration type includes Racemi [19], VMware vConverter, VMware vReplicator, VMware Site Recovery Manager (SRM) [20], and Rackware [21]. Post-configuration may involve use of eVault [22] or IBM Tivoli Storage Manager (TSM) [23], further increasing the complexity and number of choices at each step of the migration process [24, 25]. Furthermore, the migration process never executes seamlessly in a sequential manner, due to many unanticipated events at each step [26]. For example, at the discovery stage, there may be an issue with the accuracy of the discovered data. The change in scope, at the analysis step may delay further execution. The pre-configuration may be delayed due to network circuits not being ready. During migration, the source environment may have changed or secure WAN may not be available. In summary, the IT environments are highly dynamic, and the initially discovered infrastructure may no longer be accurate at the time of migration execution.

As the target environment becomes more diversified, the complexity on deciding where to move rather than how to move renders migration analytics a harder problem. The possible choices for target environments include container, (public/private) virtual machine, baremetal, POD (Performance Optimized Data Center), datacenter, geo-location, cloud provider, service model. As shown in Figure 1 that illustrates selectable cloud model, this collection can be represented as a hierarchical structure. Migrating enterprise data centers really means moving all these layers. In this paper, we focus on migration analytics below the data center level, and the objective is to see how effectively we can plan on the migration in terms of target resources. The ultimate question to answer in this paper is how efficiently we can run enterprise data centers in the cloud environments.

III. Migration Analytics

Before discovering the source environment, we have to define transformation metrics in order to identify right system attributes to discover, and also collect the target catalogs. Then with the discovered data, the analytical model aims to find the optimized resource planning. This section identifies transformation metrics for the source discovery, and propose methods to decompose servers, find matched target resources, and consolidate servers.

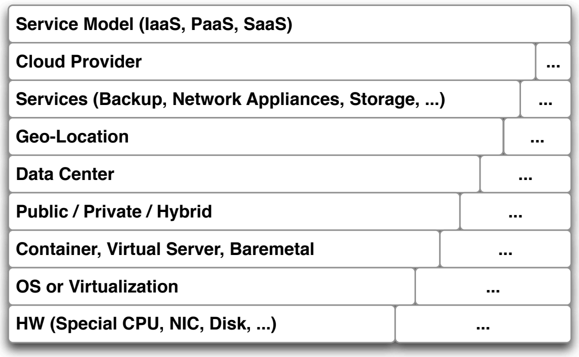
Service Model (IaaS, PaaS, SaaS)
Cloud Provider
...
Services (Backup, Network Appliances, Storage, ...)
...
Geo-Location
...
Data Center
...
Public / Private / Hybrid
...
Container, Virtual Server, Baremetal
...
OS or Virtualization
...
HW (Special CPU, NIC, Disk, ...)
...

Fig. 1. Hierarchical representation of selectable cloud resources (IaaS = infrastructure-as-a-service, PaaS = platform-as-a-service, SaaS = software-as-a-service)

A. Transformation Metrics

Transformation metrics are important system attributes that are used to transform or migrate one form of resources into new form of other resources. For example, when we want to transform a physical machine with Windows 2008 operating system and DB2 into a virtual machine running on a hypervisor, we must know computing resources (e.g., # cores, memory size, disk size), machine architecture or disk type, and more. In order words, considering transformation metrics naturally leads to the question of what to discover from the source environment. Table I lists attributes that should be considered when transforming resources for each server.

Collecting data can be done by various methods depending on operating systems. One can find most of attributes through the system files, registry, or default tools. For example, if one is using Linux or similar operating systems, shell is the easiest and the most convenient way to program and run. /proc directory contains system attributes including resources and part of physical architecture. Many standardized tools can provide lots of information as well. To name a few, *ps* reports a snapshot of the current processes, *df* reports filesystem disk space usage, *nm* lists symbols from object files, *objdump* displays detailed information from object files, *readelf* displays information about ELF object files, *lspci* displays information about PCI buses in the system and devices connected to them, *lsof* provides a list of all open files belonging to all active processes, *ldd* prints the shared libraries required by each program or shared library specified on the command line, *strace* traces system calls and signals, *ltrace* traces library call, and *netstat* prints network connections, routing tables, interface statistics, masquerade connections, and multicast memberships. In Windows systems, we can make use of vbscripts to identify the similar system attributes.

Servers are interconnected each other through network in order to provide a service. For example, a web service may need servers that run web servers, databases, file systems, or memory cache servers. This distributed architecture needs to be taken into consideration in order to capture the placement of servers. The inter-server attributes can help make decisions on the server decomposition and consolidation. To collect the inter-server data, we can look at the network statistics derived from *netstat* information, gather network ports for the communication, and infer the data sharing between servers.

Category	**Items**
Resources	# cores, memory size, disk size, network throughput
Software	libraries, softwares, legacy proprietary (non-standard) softwares, operating systems
Performance	avg/max cpu usage, max memory usage, avg/max disk usage, avg/max network usage (also measure cpu/memory/disk/network intensity with average values)
Machine architecture	cpu architecture (x86_64) and speed, multi-process, multi-thread, network accessible or library accessible, architecture dependency (x86, hyper-threading, multi-processor machine, special hw), OS dependency
Network	open ports, # interfaces
Security	sensitive data, misconfiguration, network security (firewall rules)
Management	monitoring, patching, compliance checker

TABLE I. TRANSFORMATION METRICS FOR EACH SERVER

Finally to help make decisions on server decomposition and consolidation, we need to identify versions such as software/operating system versions, and library versions, and performance characteristics such as cpu/memory/disk/network intensity. Since performance attributes are observed for a period of time, the performance characteristics of servers also can be determined based on the observed data. Version information resolves the software conflicts for installing softwares that share the same operating system or libaries. For instance, if an application uses a python module *twisted*, it requires to run on python 2.x because python 3.x version does not yet support the *twisted* module. The performance characteristics also are very useful to determine the server consolidation. For instance, if two servers have cpu intensive characteristics, they should be ruled out from the consolidation candidates because the applications inside the two servers may conflict to get cpu resources to run the processes.

Figure 2 illustrates an analytical model architecture, including collecting user preferences, discovered data and target catalogs, matching resources from source to target, consolidating servers, and decomposing a server. This is an iterative process to find the best resource matching in order to design the optimum target cloud. The resource planning step scrutinizes various factors including performance metrics, system properties, and affinity data in order to decide the best resource allocation for each server. It iteratively decides whether or not consolidating servers into one server or decomposing a server into multiple servers (or containers). Details of each process are explained in the following sections.

B. Target Catalogs

As shown in Figure 1, cloud users can choose various aspects in clouds depending on the enterprise requirements and types of workloads. Cloud users often choose one or more cloud providers depending on the service model (IaaS, PaaS, SaaS), geo-locations, and data centers. Once these high-level attributes are determined, the next step is to use the migration analytics to make the migration plans. In order for the migration analytics to accurately refect the target cloud environment, we have to understand the target catalogs first. The cloud catalogs include supported capabilities such as cloud

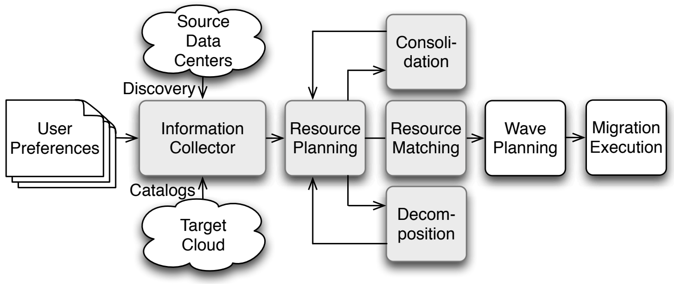

Fig. 2. Analytical model architecture

types (private, public, hybrid), architecture (container, virtual server, baremetal server), resource types (# cpu, memory size, disk size, network throughput), services (block/object storage, network appliances/connectivity, backup), software (operating systems, database softwares, other addons), and management capabilities (monitoring and manageability such as reboot). Eventually we map the source resources to the optimal target resources defined in the catalogs.

Since cloud providers support different catalogs, we need to identify the details of those catalogs. Table II summarizes the example catalogs from IBM SoftLayer. We can also easily collect the supported catalogs from PaaS or SaaS, and also from other cloud providers if cloud users decide to use one of them. The catalogs are used as an input to the migration analytics when it seeks matched target resources from observed source resources.

Categories	**Offerings**
# Data centers	49 (as of 2015)
Cloud Type	private/public/hybrid
Architecture	virtual server (VS), baremetal server (BS)
Maximum resources	VS: 8 cores, 64GB memory, 8100GB disk, 1Gbps network, BS: 40 cores, 512GB memory, 97TB disk, 10Gbps network throughput
Software	OS: CentOS, Citrix, CloudLinux, Debian, FreeBSD, Microsoft, OSNEXUS, Parallels, Redhat, Ubuntu, VMware, Addons: Citrix essentials, idera backup, Microsoft WebMatrix, vCenter, control panel: cPalen/WHM, parallels plesk panel, Database: MongoDB, Riak, Cloudera, MSSQL, MySQL, Anti-Virus: McAfee VirusScan, Intrusion detection & protection: McAfee host intrusion protection w/reporting, Monitoring: monitoring package
Management	Monitoring: host ping and tcp service monitoring, response: automated reboot from monitoring, insurance: business continuance insurance

TABLE II. TARGET CATALOGS FROM SOFTLAYER [27]

C. Resource Planning and Matching

Given the discovered source attributes and the target catalogs, we formulate CSP to find the optimally matched resource. CSP is defined by a set of variables, and a set of constraints. Each variable has a nonempty domain of possible values. Each constraint involves some subset of the variables and specifies the allowable combinations of values for that subset. A state of the problem is defined by an assignment of values to some or all of the variables. An assignment that does not violate any constraints is called a consistent or legal assignment. A complete assignment is one in which every variable is mentioned, and a solution to a CSP is a complete assignment that satisfies all the constraints possibly with an objective function [28].

For servers, we define selected transformation metrics as variables, $S_i = \{s_1, s_2, ..., s_n\}$, where s is a transformation metric, and n is the number of transformation metrics. We denote $T_i = \{t_1, t_2, ..., t_n\}$ as a matched cloud resource from a server S_i. In order words, $T_i = T(S_i)$, where $T(\cdot)$ is a transformation function that satisfies a complete assignment from the CSP. The domain is defined as $D = \{d_1, d_2, ..., d_m\}$, where d is an item of the target catalog, and m is the number of entire items in the catalog. Basically, each cloud resource t is one from D. Lastly, the constraints are defined as follows:

- For each assignment, a resource type should be equal between s and d.
- For the numerical resources such as # cores, memory size, disk size, or network throughput, $s \leq d$.
- For other resources such as operating system/version, or libraries, $s = d$.

We use a lookahead algorithm to forward check the constraints between the current and past variables and the future variables. With the forward checking, we can immediately know the current partial solution is inconsistent, when the domain of a future variable becomes empty. In this case, either another value for the current variable is tried or the algorithm backtracks to the previous variable. Then the state of the domains of future variables is restored as they were before the assignment which led to failure, This can not be achieved with a simple backtracking algorithm [29].

One potential pitfall of the migration is that the migration engineers tend to think the source resources are optimized to run the existing applications. However from the sample performance monitoring data shown in the Figure 3 from the real enterprise datasets, we can easily verify the maximum usage of the resources are not close to the allocated capacity. The cloudish approach is to assign the maximally required resource at any given time, and scale applications elastically when the usage increases and the better performace is required. Therefore after finding the resources, we adjust the resources based on the maximum performance used during the period of time. For example, the souce machine with 4 cores (400%) runs CPU in average 32%, and maximum 120%, we subtract 2 cores, thus assign 2 cores.

In case of being unable to find the matched numerical resources, the server is marked as *uncloudable*, and if s is not supported in the catalog and can manually be installed under the baremetal server, the labor cost is added for that s. This means we do not rule out s because it is not supported in the cloud catalog if s can be supported anyhow. This is because even though we aim to automate the entire transformation process, there are still pieces that can not be supported automatically, but the model should take into consideration for the cost minimization. Since the target cloud offers a pay-as-you-go model, the expenditure spent on the cloud resources

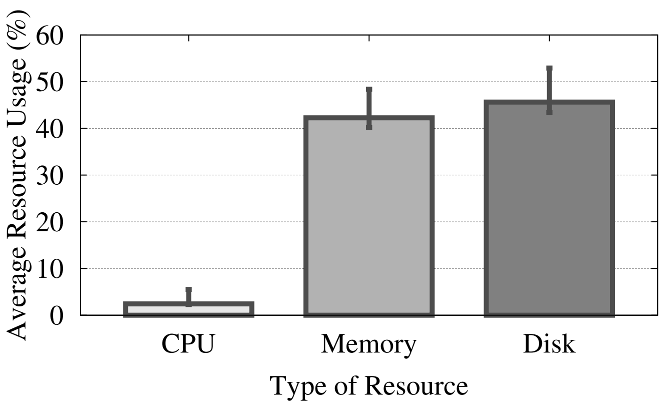

Fig. 3. Average resource usage from a customer's data center (335 servers, 1 week)

needs to be minimized. We define $Cost(T_i)$ is the total cost for the server i against the target cloud, and the objective function is:

$$minimize \sum_{i=1}^{N} Cost(T_i) \quad (1)$$

When the number of servers is N, the number of transformation metrics is n, and the number of catalog items is m, we can solve this problem in a polynomial time, $O(Nnm)$, by iterating each item.

D. Server Consolidation

One of the primary benefits of cloud environment is to reduce operational expenses by paying only for the resources that are used, called a pay-as-you-go model [2]. Again, this is the same problem, if servers in the source environment are migrated intact, requesting the same memory and CPU capacity, this will likely result in a very large unnecessary expenditure. For example, if a source server contains 64 GB of memory, but uses only 4 GB of it, there is no need to request the full 64 GB in the target environment. Doing so would result in underutilized servers in the target that consume extra energy and space. In addition, it may be possible to consolidate servers into one target virtual machine in order to reduce the overhead and cost of deploying multiple individual ones.

However, not all servers are good candidates for consolidation. The transformation metrics shown in Table I need to be checked for any conflicts between servers. Two servers offering services on the same port, for example, would not be able to move to the virtual machine and use the same interface. Similarly, if the machines are running two different operating systems or heavily rely on the same applications, they should not be consolidated. Furthermore, servers that have the resource (i.e., CPU intensive) should not be consolidated into a server. Also, care should be taken to ensure that resource-intensive servers are placed on separate target servers in order to prevent a degradation of service due to an overloaded server. The problem of virtual machine consolidation has been studied mostly in the context of energy efficiency [4] and for consolidating virtual machines in the minimal number of physical servers. Used in the migration space, this approach has the potential to reduce operational expenses as well by reducing the number of source physical servers that need to be mapped to target hosts.

Consolidating applications into one virtual machine or physical machine, or merging them into a new environment requires the conflict resolution:

- No network ports should be used in common. $P(S_i) \bigcap P(S_j) = \{\emptyset\}$, where $P(\cdot)$ is a list of network ports used. If an administrator can change the default ports manually, this constraint can be passed.
- Performance intensity should be different. For example, two applications that have a CPU intensity should not be considered to merge into the one server. $I(S_i) \neq I(S_j)$, where $I(\cdot)$ is the performance intensity of the server (i.e., CPU, memory, disk, or network).
- Library and operating system versions applications run on should be supported. $V(S_i) = V(S_j)$, where $V(\cdot)$ a list of operating system/versions and library versions that are used by server applications.

E. Server Decomposition

Many applications, if not designed for distributed deployment, tend to run on the same server because they do not have any conflicts on the configuration such as network ports, and it is convenient for administrators to have them managed in the same server. Also in the perspective of performance, applications run without network delay, which can incur performance degradation. However this often brings troubles because applications running on the same server can use the same resource simultaneously, and this can result in slowing down each other. Not only the performance issues, the scalability is an another issue because, for example, the database can not easily be scaled out when it is running in the localhost, together with a consumer application.

When an administrator wants to redesign application deployment to increase performance and support an elastic scalability in the target cloud, a server that runs multiple networked applications needs to be decomposed into separate components. For example, a high-end physical server that runs many applications can be decomposed into many virtual machines, each of which has an application, and the decomposed components are configured to communicate through the network. Another good example may be a container-based deployment for supporting the dev/ops life cycle with packaged applications.

Decomposing a server into multiple pieces can be an easy task if we can identify all the networked applications, and how much resources each application requires. We check each component in the server S_i, whether or not s_j runs independently through the network and communicates with toher applications using an API. Also, we use the utilities such as *top* and *Windows Task Manager* to gather further information. The application performance is important to decide how much resource it needs and what platform it should run on. Obviously, it is a bad practice to run a multi-process application only in one container.

After decomposing a server into multiple pieces, once again each piece should find a matched resource from the cloud catalogs as shown in the previous section III-C. This is an iterative process that goes back to the resource matching again to find the optimal resource with a minimum cost.

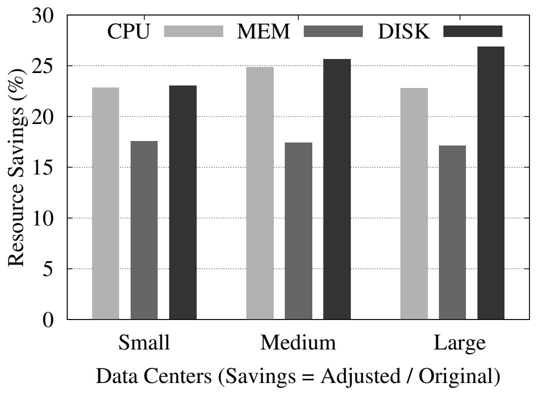

Fig. 4. Savings for Itemized Resources

IV. Evaluation

We evaluate the analytical model to test its validity and demonstrate how it can effectively provide resource transformation/consolidation/decomposition. Particularly, we focus on the following goals to check:

- resource planning effectiveness with performance metrics (§IV-B),
- resource matching ratio between the planned resources and target catalogs (§IV-C),
- ratio of potential server consolidation (§IV-D), and
- ratio of potential server decomposition (§IV-E).

A. Datasets

To validate the variability of different datasets, we use three different sizes of data cetners as shown in Table III: a small size data center, a medium size data center, and a large size data center. The datasets are composed of different operating systems and the virtualization ratios (running on virtual machines) are also varied. As enterprise customers, the VMware hypervisor (ESXi) is the default commercial platform. But since we do not consider the hypervisor as a metric, any hypervisor can be a tool to virtualize servers. The collected data include metrics in Table I at the minimum and more such as server affinity. Performance items are measured for two weeks period.

For target catalogs, we use one cloud provider, SoftLayer, in the experiments, but the same experiment can be easily performed to any other cloud providers as well. The catalog details are shown in Table II. When certain operating systems or softwares are not supported in virtual servers, we use baremetal machines with VMware hypervisors to run virtual servers with custom installations. This allows the planning engine to migrate operating systems such as Solaris, AIX, and HP-UX to the target environment without being tied to cloud service offerings.

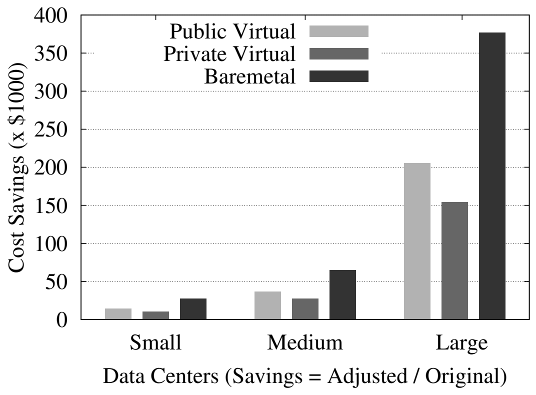

Fig. 5. Cost Savings with Resource Adjustment

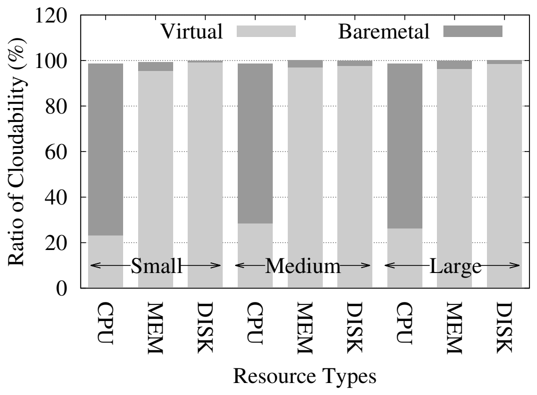

Fig. 7. Itemized Cloudability Ratio

To note, there are already prior works for the resource consolidation using Knapsack or Bin-Packing algorithms [4, 30], but this paper does not aim to provide better algorithms to achieve the same goal. Instead, this work aims to help enterprises to reason about how to plan on the target environment with real datasets. Also, we assume that physical servers and virtual servers convertible, meaning we can migrate a physical server into a virtual server (technically we can easily do this using one-to-one migration tools such as VMware vConverter).

Name	# Servers	Operating Systems	Virtualized
SMALL	212	Windows (75%), Solaris (25%)	62.7% (VMware)
MEDIUM	543	Windows (65%), Linux (18%), Solaris (10%), AIX (5%), HP-UX (2%)	8.3% (VMware)
LARGE	2012	Windows (45%), Linux (30%), Solaris (18%), HP-UX (7%)	17.6% (VMware)

TABLE III. SPECIFICATIONS OF DATA CENTERS USED FOR CLOUD ANALYTICS

B. Resource Planning

When planning resources, after discoverying, an immediate step is to identify whether resources can be adjusted at the target environment. This leads to the question whether or not we can save expenses without sacrificing performance. To answer this question, we scrutinize both performance and system properties to find out how much overprovisioned resources are. The main objective is shown in Equation (1) that minimizes the overall costs through the resource planning. As shown in Figure 4 that illustrates savings for itemized resources, resources are often overprovisioned for the readiness of the worst case scenarios. However since the cloud computing is meant to provide elastic scalability when the request rate increases, a traditional provisioning strategy that overprovisions resources only charges more operational expenses. As a result, we can significantly save operational expenses by adjusting resources. For CPU and DISK resources, we can save around 20% - 25% or more, but for memory resources, we can only save around 16%. Since the memory is known to be a resource bottleneck, this is not the surprising result.

Although the ratio of resource savings in Figure 4 for three data centers look similar, the absolute values of them are completely different in that sizes of data centers become significant when we actually calculate the real monetary savings from the resource adjustment. Figure 5 illustrates the operational cost savings with resource adjustment. Since there are different kinds of target environments (i.e., public/private virtual and baremetal) and their monthly charges are also different, we show costs for different target environments. The savings of the large size data center for baremetal environment is 14 times more savings than the small size data center. This is because the baremetal can consolidate more virtual servers in one physical server. Since the resource adjustment is based on the maximum resource usage in its peak time, we do not expect to see any performance degradation due to the resource adjustment.

The overall potential resource adjustment ratios and cost savings are significant, yet we still want to look deeper ramification through the distribution function on the resource adjustment. Figure 6 illustrates itemized cumulative density function (CDF) for each resource type. The number of cpu core savings are quite uniformly distributed below 40, which means the cpu cores are normally overprovisioned throughout all the machines. Disk also shows a similar distribution as the number of cpu cores. However the cdf of memory size shows 20% - 30% machines largely underutilize the memory resources.

C. Resource Matching

Once the resource planning is made by adjusting the resources based on the performance and cost analysis, it is time to verify those resources can fit into the cloud environment using the target catalogs. In the custom virtual environment using a baremetal server and own hypervisor, allocating maximum resources is allowed. However the cloud providers usually do not expose the maximum capacity, instead it standardizes the resources with limited sizes. Figure 7 illustrates the ratio of cloud-fitness (here called cloudability). The CPU cloudability is only around 20% when migrating into virtual servers, which means that servers can not fit into virtual servers that cloud providers offer due to the oversized CPU. However baremetal servers usually do not have the limitations on provisioning resources so that almost all of servers can fit. Still, there are some servers that require extremely high performance. We notice some custom-ordered machines can not usually be supported at the target environment. Most of memory and disk

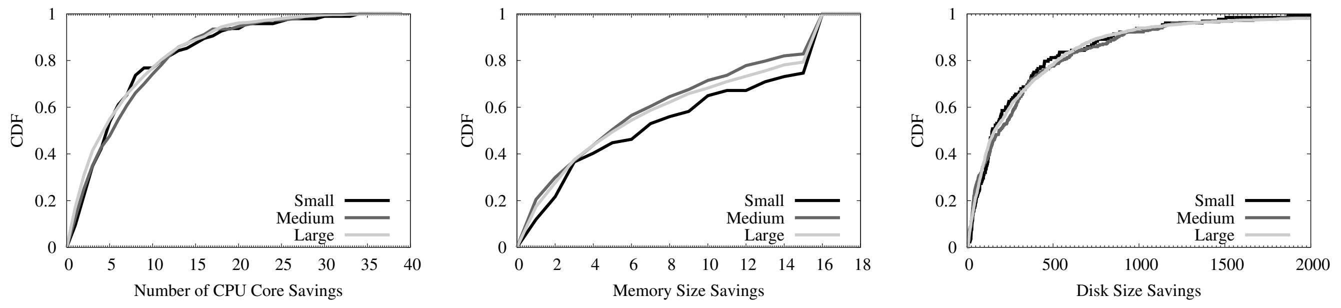

Fig. 6. CDFs for Cost Savings of Different Resources

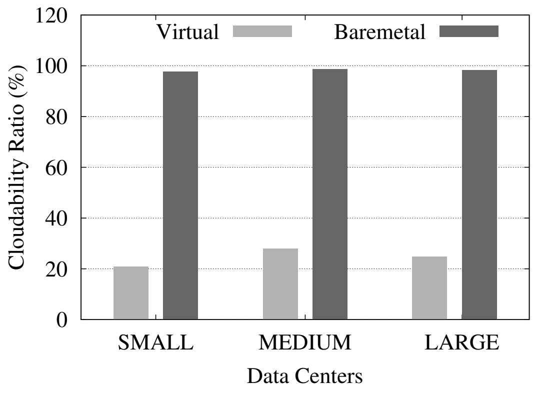

Fig. 8. Total Cloudability Ratio

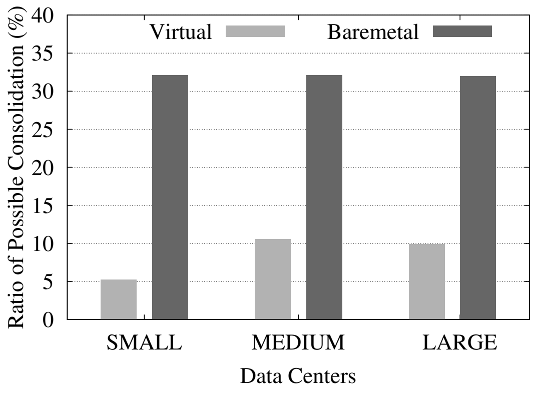

Fig. 9. Server Consolidation

resources can fit into virtual servers because memory and disk can be extended to have more capacity. Especially, the storage area network (SAN) can be easily expandable.

As shown in Figure 8, the total cloudability for virtual servers is limited by the CPU. This tells us that if users want to migrate into the virtual server environment, they need to redesign the application deployment using elastic scalability. One simple example is that a load-balancer can scale out services by distributing service requests. Another example can be adding a memory caching layer in order to compensate the database performance.

D. Server Consolidation

As explained in Section III-D, applications or services can be consolidated into a server for the sake of reducing operational expenses if their performance does not degrade. To decide the consolidation, conflicts need to be resolved in order not to disrupt services. First of all, resource intensities of applications should not be overlapped. For instance, two applications that are CPU intensive should not be consolidated into one server because they will definitely fight for more CPU resources, meaning the performance degradation. Next, the system properties should be disjoint. For instance, if two services use the same network port, they can not run on one server. The issuses with system properties can be easily solved by changing the application configuration, though. The consolidated applications should have the same security level in order to be managed in the same way. In the management perspective, the unit of the security management is normally per server.

Through comparison between servers that satisfy the requirements, we find the potential number of server pairs that can be consolidated into one server. From our experiments, we find that there are no more than two servers that can be consolidated into a server due to conflicts. So the number of servers that consolidated into a server is always two in our experiments. Figure 9 that illustrates the ratio of consolidatable servers. If we target virtual servers, the ratio of consolidation is around 5% - 10%, but for baremetal servers, the ratio becomes above 30% due to larger capacity.

E. Server Decomposition

As explained in Section III-E, a server can be decomposed into multiple server instances in order to provide the distributed deployment. The main reason we may consider the decomposition is that the cloud-like application deployment requires modularity to achieve an elastic scalability when the ratio of service requests changes over time. So the condition for the server decomposition is that applications or services should be able to run as modules through network connections. Through the decomposition, we expect more manageability and improved performance. As depicted in Figure 10 illustrates the ratio of potential decompositions, the ratio of server decomposition is very high, that is 60% servers can be decomposed into multiple servers for virtual servers. For baremetal servers, it can be easily speculated higher than the virtual server case.

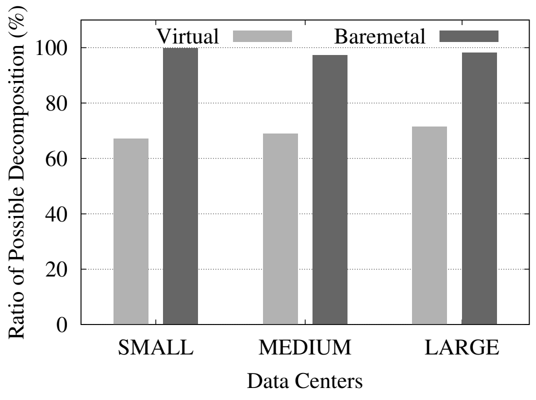

Fig. 10. Server Decomposition

This is mainly because most of applications can be run as modules with network connectivity.

V. Related Work

There are many research activities that work on the migration analytics. Hajjat et al. [2] analyze the potential benefits of hybrid cloud deployments of enterprise applications, and the importance and feasibility of a planned approach to making migration decisions. Also, authors have shown the feasibility of automatic and assurable reconfiguration of reachability policies as enterprise applications are migrated to hybrid cloud models. Zhang et al. [26] propose an analystics to automatically detect software misconfigurations before the migration. Bai et al. [15] propose an analytical model that can discover the complex server-to-server and application-to-server relationships. We complement these works to extend the resource planning, server consolidation, and server decomposition as a part of the migration analytics.

The end-to-end migration orchestration has been researched. Liu et al. [17] propose COPE (Cloud Orchestration Policy Engine), a distributed platform that allows cloud providers to perform declarative automated cloud resource orchestration. In COPE, cloud providers specify system-wide constraints and goals using COPElog, a declarative policy language geared towards specifying distributed constraint optimizations. Menzel et al. [10] present a CloudGenius framework, which automates the decision-making process based on a model, factors and QoS (quality of service) specifically for Web server migration to the Cloud. Hwang et al. [5] propose a comprehensive migration framework that spans an end-to-end migration process from discovery to post-migration configuration. Our work can be a part of this migration orchestrator to plan a migration resource transformation.

Efficient migration technologies have been studied. Al-Kiswany et al. [12] propose a migration optimization including data deduplication and prioritization of data transfers and system support to accelerate virtual machine and application startup. Wood et al. [31] propose an optimized support for live WAN migration of virtual machines that minimizes the cost of transferring storage and virtual machine memory during migrations over low bandwidth and high latency Internet links. Our work can make use of these migration technologies to migrate images efficiently.

VI. Conclusion

The enterprise-scale migration analytics provides an effective migration planning capability that can transform resources from on-premise data centers to target clouds. We have described a model to tackle the migration challenges that transform one resource type into same or another resource type in hybrid clouds. We formulate the problem as a constraint satisfaction problem, and iteratively decompose the server components and consolidate the servers. The experimental results showed significant resource savings, in turn savings for operational expenses. Also, the possible server consolidation and decomposition allow enterprises to plan better by knowing possible resource efficiency. We think the work can be used as a part of the migration workflow and it can impact on the overall migration resource optimization. Our future work include extending the migration analytics into the consumable software as a service model as a part of the migration orchestrator.

References

[1] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. A view of cloud computing. *Commun. ACM*, 53(4):50–58, 2010.

[2] Mohammad Hajjat, Xin Sun, Yu-Wei Eric Sung, David Maltz, Sanjay Rao, Kunwadee Sripanidkulchai, and Mohit Tawarmalani. Cloudward bound: Planning for beneficial migration of enterprise applications to the cloud. *SIGCOMM Comput. Commun. Rev.*, 40(4):243–254, 2010.

[3] Xin Meng, Jingwei Shi, Xiaowei Liu, Huifeng Liu, and Lian Wang. Legacy application migration to cloud. In *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, pages 750–751, July 2011.

[4] M. Marzolla, O. Babaoglu, and F. Panzieri. Server consolidation in clouds through gossiping. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a*, pages 1–6, June 2011.

[5] Jinho Hwang, Yun-Wu Huang, Maja Vukovic, and Nikos Anerousis. Enterprise-scale cloud migration orchestrator. In *IFIP/IEEE IM 2015 Symposium*, 2015.

[6] Jinho Hwang, Yun-Wu Huang, Maja Vukovic, and Jill Jermyn. Cloud transformation analytics services (a case study of cloud fitness validation for server migration). In *IEEE International Conference on Services Computing (SCC)*, June 2015.

[7] A. Khajeh-Hosseini, I. Sommerville, J. Bogaerts, and P. Teregowda. Decision support tools for cloud migration in the enterprise. In *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, pages 541–548, July 2011.

[8] Jill Jermyn, Jinho Hwang, Kun Bai, Maja Vukovic, Nikos Anerousis, and Salvatore Stolfo. Improving readiness for enterprise migration to the cloud. In *Proceedings of the Middleware Industry Track*, Industry papers, pages 5:1–5:7, New York, NY, USA, 2014. ACM.

[9] Mike Nidd, Kun Bai, Jinho Hwang, Maja Vukovic, and Michael Tacci. Automated business process discovery. In *IM'15*, IM'15, 2015.

[10] Michael Menzel and Rajiv Ranjan. Cloudgenius: Decision support for web server cloud migration. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 979–988, New York, NY, USA, 2012. ACM.

[11] H. Teyeb, A. Balma, N. Ben Hadj-Alouane, and S. Tata. Optimal virtual machine placement in large-scale cloud systems. In *Cloud Computing (CLOUD), 2014 IEEE 7th International Conference on*, pages 424–431, June 2014.

[12] Samer Al-Kiswany, Dinesh Subhraveti, Prasenjit Sarkar, and Matei Ripeanu. Vmflock: Virtual machine co-migration for the cloud. In *Proceedings of the 20th International Symposium on High Performance Distributed Computing*, HPDC '11, pages 159–170. ACM, 2011.

[13] Birgit Pfitzmann and Nikolai Joukov. Migration to multi-image cloud templates. In Hans-Arno Jacobsen, Yang Wang, and Patrick Hung, editors, *IEEE SCC*, pages 80–87. IEEE, 2011.

[14] Ali Khajeh-Hosseini, David Greenwood, and Ian Sommerville. Cloud migration: A case study of migrating an enterprise it system to iaas. In *Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing*, CLOUD '10, pages 450–457, Washington, DC, USA, 2010. IEEE Computer Society.

[15] Kun Bai, Niyu Ge, H. Jamjoom, Ea-Ee Jan, L. Renganarayana, and Xiaolan Zhang. What to discover before migrating to the cloud. In *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*, pages 320–327, May 2013.

[16] Jing Tai Piao and Jun Yan. A network-aware virtual machine placement and migration approach in cloud computing. In *Grid and Cooperative Computing (GCC), 2010 9th International Conference on*, pages 87–92, Nov 2010.

[17] Changbin Liu, Boon Thau Loo, and Yun Mao. Declarative automated cloud resource orchestration. In *Proceedings of the 2Nd ACM Symposium on Cloud Computing*, SOCC '11, pages 26:1–26:8, New York, NY, USA, 2011. ACM.

[18] R. Filepp, L. Shwartz, C. Ward, R.D. Kearney, K. Cheng, C.C. Young, and Y. Ghosheh. Image selection as a service for cloud computing environments. In *Service-Oriented Computing and Applications (SOCA), 2010 IEEE International Conference on*, pages 1–8, Dec 2010.

[19] Racemi. http://www.racemi.com/.

[20] VMware. http://www.vmware.com/.

[21] Rackware. http://www.rackwareinc.com/.

[22] eVault. https://www.evault.com/.

[23] IBM Tivoli Storage Manager. http://www-03.ibm.com/software/products/en/tivostormana.

[24] Eric Keller, Soudeh Ghorbani, Matt Caesar, and Jennifer Rexford. Live migration of an entire network (and its hosts). In *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*, HotNets-XI, pages 109–114. ACM, 2012.

[25] C. Ward, N. Aravamudan, K. Bhattacharya, K. Cheng, R. Filepp, R. Kearney, B. Peterson, L. Shwartz, and C.C. Young. Workload migration into clouds challenges, experiences, opportunities. In *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, pages 164–171, July 2010.

[26] Jiaqi Zhang, Lakshminarayanan Renganarayana, Xiaolan Zhang, Niyu Ge, Vasanth Bala, Tianyin Xu, and Yuanyuan Zhou. Encore: Exploiting system environment and correlation information for misconfiguration detection. In *Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '14, pages 687–700. ACM, 2014.

[27] IBM SoftLayer. http://www.softlayer.com/cloud-servers.

[28] Miguel A. Salido, Adriana Giret, and Federico Barber. Distributing constraints by sampling in non-binary csps. In *In IJCAI Workshop on Distributing Constraint Reasoning*, pages 79–87, 2003.

[29] Sally C Brailsford, Chris N Potts, and Barbara M Smith. Constraint satisfaction problems: Algorithms and applications. *European Journal of Operational Research*, 119(3):557–581, 1999.

[30] Susheel Thakur, Arvind Kalia, and Jawahar Thakur. Server consolidation algorithms for cloud computing environment: A review. *International Journal of Advanced Research in Computer Science and Software Engineering*, 3(9):379–384, 2013.

[31] Timothy Wood, K. K. Ramakrishnan, Prashant Shenoy, and Jacobus van der Merwe. Cloudnet: Dynamic pooling of cloud resources by live wan migration of virtual machines. In *Proceedings of the 7th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments*, VEE '11, pages 121–132, New York, NY, USA, 2011. ACM.