IBM Research Report

ReCon: A Tool to Recommend Dynamic Server Consolidation in Multi-Cluster Data Centers

Sameep Mehta
IBM Research Division
IBM India Research Lab
New Delhi - 110070. India.

Anindya Neogi
IBM Research Division
IBM India Research Lab
New Delhi - 110070. India.

IBM Research Division
Almaden - Austin - Beijing - Delhi - Haifa - T.J. Watson - Tokyo - Zurich

Abstract

Renewed focus on virtualization technologies and increased awareness about management and power costs of running under-utilized servers has spurred interest in consolidating existing applications on fewer number of servers in the data center. The ability to migrate virtual machines dynamically between physical servers in real-time has also added a dynamic aspect to consolidation. However, there is a lack of planning tools that can analyze historical data collected from an existing environment and compute the potential benefits of server consolidation especially in the dynamic setting. In this paper we describe such a consolidation recommendation tool, called ReCon. Recon takes static and dynamic costs of given servers, the costs of VM migration, the historical resource consumption data from the existing environment and provides an optimal dynamic plan of VM to physical server mapping over time. We also present the results of applying the tool on historical data obtained from a large production environment.

1 Introduction

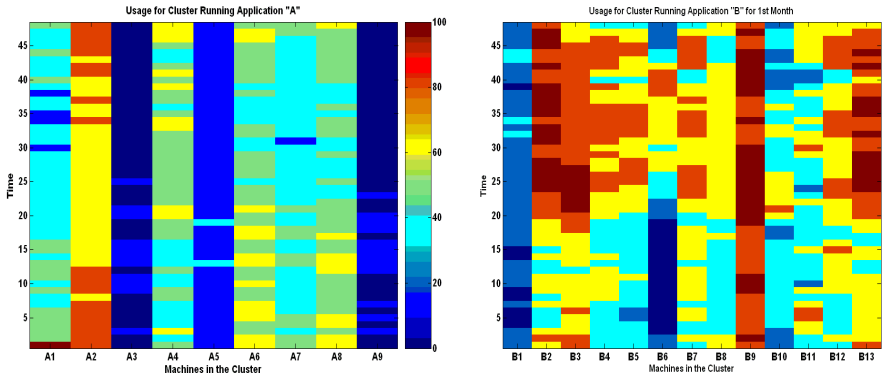
Virtualization technologies first appeared in the 1960s to enable timesharing of expensive hardware between multiple users. As hardware became cheaper, virtualization gradually lost its charm. However, since the late 1990s there has been a resurgence of these technologies. In server environments it has gained such a huge momentum that in a couple of years almost every machine shipped is expected to support virtualization in one or more layers of the hardware, firmware, and software stack.

Server virtualization has regained popularity for various reasons. Virtual machines support more flexible and finer grain resource allocation and configuration than physical machines. Even though the hardware cost of servers has dropped, the cost of management and TCO of these servers has gone up drastically. It has been shown that the cumulative running management cost of a server over its lifetime is significant when compared to its hardware cost. Virtualization enables consolidation of a number of smaller servers as virtual machines on a larger server. The assumption is that the management cost does not scale up with the number of virtual machines. Also a single set of management controls can now be used to manage the virtual machines on a server instead of individual controls for every server. Data center studies show that the lower end servers, perhaps because of the lower cost, are often run at lower utilization compared to higher end servers. Thus consolidation using virtualization leads to more efficient utilization of hardware resources. Fewer number of heavily utilized servers also leads to savings in expensive floor space and facilities management costs. Finally, complex virtualization software tends to hide the heterogeneity in server hardware and make applications more portable or resilient to hardware changes.

A key issue in virtualization is thus to map an existing or fresh workload to virtual machines on physical servers. In a completely new setup, average benchmarked numbers from servers can be used to plan the virtualized environment. However, for an existing data center such a consolidation exercise is easier said than done. In our experience, often customers are very reluctant to move to a virtualized environment from their existing system as they are not convinced about the consolidation related benefits of virtualization. Even though server handbooks and static planning spreadsheets may promise that the existing servers can be clubbed into a fewer number, it needs a more detailed workload study in the actual environment and demonstratable evidence of the benefits based on historical data collected from the same environment.

We have developed a tool called *ReCon* that uses historical resource usage monitoring data from the servers in an environment and recommends a dynamic consolidation plan on the existing set of servers or a different target set. As opposed to existing tools, *ReCon* does not stop at providing a static consolidation plan based on the average resource utilization numbers. It actually assumes live virtual machine migration capabilities in the virtualization platform to recommend a dynamic consolidation pattern where virtual machine to physical machine mappings can be simulated over time to demonstrate the real benefits of embracing virtualization with migration.

Figure 1(a)-(b) shows the CPU utilization patterns for two clusters in a large data center averaged over the entire month. Colors towards "red" represent high utilization and colors closer to "blue" represent low utilization. It is clear that in Cluster A, some of the machines are grossly under-utilized throughout the day. Therefore, the workloads on these machines can be consolidated statically and the number of machines can be reduced permanently in the setup. Cluster B and some of the machines in Cluster A show that there are long low utilization periods during the day (not for entire day). This indicates that even though the number of machines cannot be reduced permanently, it is possible to consolidate the workloads on fewer machines and switch off the remaining servers or put them in some low power state during those periods. Of course, it entails using virtual machine



(a) (b)

Figure 1: Heatmaps showing average intra-day variation in CPU utilization of two clusters in a single month. Each column corresponds to a physical server. Th y-axis is divided into 48 30min intervals in a day. Each 30 min sample for a server is averaged for the same time interval over the whole month.

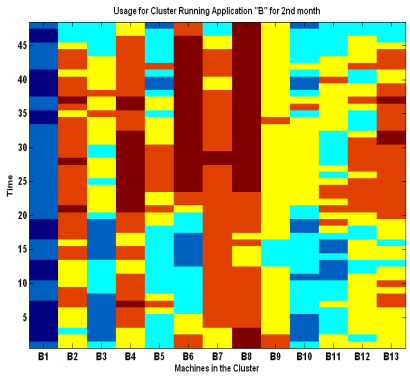
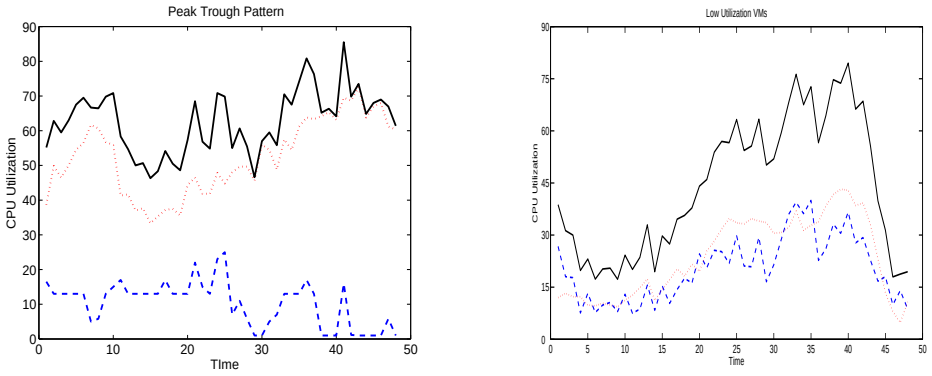


Figure 2: Heatmaps showing average intra-day variation in CPU utilization of two clusters in a single month. Each column corresponds to a physical server. Th y-axis is divided into 48 30min intervals in a day. Each 30 min sample for a server is averaged for the same time interval over the whole month.

migration technologies to migrate the applications at runtime. Figure2(c) shows the utilization pattern of cluster B for the next month. It is evident from the figures that the utilization changes over time. For example B7 and B8 are used moderately in first month however, heavy usage is seen in second month. On the other hand B9 shows opposite behavior. To handle such cases the reconsolidation tool should be able to capture dynamism exhibited by clusters.

Even if the average machine utilizations are not very low, the utilizations on different machines are sometimes complementary in nature, i.e., when one machine in a cluster is at high utilization, another machine in the same or different cluster may be at a low utilization. This is illustrated in the set of CPU utilization plots on a pair of machines shown in Figure 3. The dotted lines are the CPU utilizations captured in a time window for two servers and the solid line is the summation of the two dotted lines. Dynamic consolidation can happen either due to periods of low resource utilization of the packed applications (Figure 3(b)) or when applications demonstrate complementary resource behavior during a period (Figure 3(a)). In the course of this discussion we have often simplified resource utilization to be captured only by CPU utilization. In practice, multiple parameters, such as memory, disk and network I/O bandwidth consumption etc., need to be considered to create a *volume* that represents resource utilization of an application workload.

In this paper, we describe the architecture and algorithms of *ReCon* and its validation using historical traces collected from a large data center. Unlike other tools that perform static consolidation based on benchmarked results, *ReCon* operates on historical data and investigates the scope of performing dynamic consolidation utilizing virtual machine migration technologies. By varying several parameters, *ReCon* allows a user to process historical resource consumption traces in various ways



(a) (b)

Figure 3: (a) Example showing peak-trough pattern (b) Example showing low utilization VMs

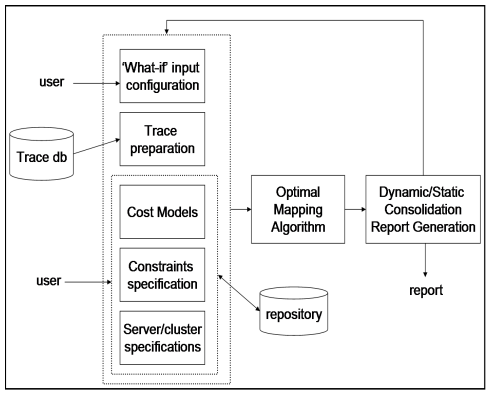


Figure 4: *ReCon* tool architecture

and generate results to investigate the entire solution space in performing server consolidation, static or dynamic, in the actual environment. For example, the user is able to find out under a given set of input parameters what is the typical number of machines to be used in a cluster, what is the scope of dynamic consolidation with complementary behavior or due to low utilization, how much of inter or intra cluster dynamic consolidation is possible etc. Armed with these experimental results on various what-if scenarios the user is now able to take a more informed decision on whether to move to a virtualized environment and what kind of dynamic consolidation behavior to expect with the real workload.

The rest of the paper is organized as follows. In Section 2 we describe the ReCon architecture. Section 3 presents the formal problem definition and describes the algorithms implemented in ReCon and Section 4 discusses the experimental validation of ReCon on actual traces. We describe some of our ongoing and future initiative in Section 6 Section 7 concludes the paper with a discussion of future work.

2 ReCon Overview

In this section we describe *ReCon* architecture and the problem statement. The detail algorithms in *ReCon* and a use case based validation is covered in the later sections.

A high level view of the *ReCon* tool is shown in Figure 4. The trace database is a set of measurements taken from the actual system under consideration. These measurements are typically in a timeseries format containing timestamped records of a CPU, memory, network, disk I/O bandwidth consumptions of physical servers in an existing system. The general approach of *ReCon* is to treat each of the servers in the existing setup (also called "source" servers) as virtual machines in a new setup. Thus the resource consumptions of the source servers are treated as the resource requirements of the corresponding VMs in the "target" servers, i.e. physical machines in the new setup. Additional information is captured from the user regarding the source and target server configurations. This is required to scale the resource requirements of the VMs when mapped to different target servers as well as ascertain the capacity of the target servers to satisfy the VM requirements. It

is possible to have the same set of source and target servers if the user wants to investigate static or dynamic consolidation using virtualization in the existing setup. The information regarding the cluster memberships of the servers is also captured so that different costs and constraints can be associated with intra and inter cluster consolidation.

The costs for servers are simply broken into two broad components, static and dynamic. The static component is used to capture the base cost of running a physical server with workload allocated to it. This cost is accounted for every target server that is powered on. In a static consolidation scenario it must have at least one virtual machine allocated to it. In the dynamic consolidation case there may be periods when the server is "on" in anticipation of future workload because there may not be ample window of opportunity to turn the server off in between allocations. Static server cost may be used to capture management, running costs etc. The dynamic cost of a server is assumed to vary with the utilization of the server. Power cost is a predominant component of this cost. Besides server costs, the user can also specify virtualization costs, such as inter and intra cluster VM migration costs.

Constraints specified in *ReCon* are essentially to restrict the space of possible mappings between VMs and physical servers. These may be system constraints, such as the CPU utilization of a target server should not exceed 90% or the network traffic due to VM migration should not be more than 10% of the LAN bandwidth, or the power cost of the data center at any point should not exceed a threshold. There may be various application level constraints, such as two applications running in two VMs cannot be mapped to the same target server in a defined time interval. There may be also legal constraints to prevent certain application VMs to be mapped to the same target server to enforce hardware isolation.

The server and cluster specifications, costs, constraints, and evaluation results can be stored in an internal repository and reused later. *ReCon* can be used as a "what-if" analysis tool. The input parameters for the analysis can be fed to the tool to process the historical trace data and generate a report. Based on the report output the user can change the parameters and review the impact on the report. Various "what-if" input parameters, costs, and constraints can be tweaked to generate consolidation reports. Input parameters include the window of dynamic consolidation, the period that a server should have no workload to consider turning it off etc. Besides the static or dynamic consolidation plan the reports essentially capture all the micro-measurements of a consolidation result. Details of these micro-measurements and the parameters to tweak can be found in the experimental evaluation section where we validate the tool in a production environment.

The core optimal mapping algorithm in *ReCon* can be used to take all the parameters, costs, constraints, configurations and process the input traces to generate static or dynamic server consolidation. For dynamic consolidation the historical data is divided into non-overlapping time windows called "consolidation windows". In each window an optimal mapping from source to target servers, i.e. VMs to virtualized physical servers is created as an incremental update from the mapping in the previous window. Runtime VM migration technologies are assumed to move from one mapping to the next. In the static consolidation case, the time window is assumed to be the entire trace and a single mapping is generated. Next, the algorithm details are discussed in detail.

3 Algorithm

Informally, the problem of recommending server consolidations can be stated as: *Given N application VMs, find n physical machines where $n < N$ such that each VM is assigned to one physical machine while satisfying domain specified constraints.* Next, we present the notations used in this section.

3.1 Basic Notations

Let $VM = \{VM_1, VM_2, \ldots, VM_N\}$ be the set of virtual machines in the current architecture. Each VM_i observes and stores K variables denoted by $\mathcal{O} = \{O_{(1,i)}, O_{(2,i)}, \ldots, O_{(K,i)}\}$. Each VM is monitored for T time steps. The time series generated by j^{th} sensor of i^{th} VM is denoted by $O_{(j,i)} = \{O^1_{(j,i)}, O^2_{(j,i)}, \ldots, O^T_{(j,i)}\}$. Also assume, that N VMs are partitioned into H groups such that each group is primarily responsible for handling one application. In an industrial environment each of these partition can be thought of as a cluster of machines. The membership of N VMs to H clusters is specified by a $N \times H$ matrix $\mathcal{M}em$, s.t. $\forall i_{1toN}, j_{1toH}$ $\mathcal{M}em(i,j) = 1$ if i^{th} VM belongs of j^{th} partition (application), 0 otherwise.

$$B = \sum_{i=1}^{N}(Cost_i + \mathcal{F}(CPU_i)) - \sum_{j=1}^{N}(Cost_j \times Y_j + \sum_{k=1}^{N} MCost_{k,j} \times A_{k,j} + \mathcal{F}(\sum_{k=1}^{N} CPU_k \times A_{k,j})) \quad (1)$$

3.2 Data Preprocessing

The monitoring tool stores the resource utilization as percentage of total resource, for e.g. 50% CPU utilization. Such percentages, however, cannot be used to establish (in)equality between two machines, e.g., 50% of 1 Ghz and 25% of 2 Ghz. Since the hardware specification of machines are readily available, we use this information to normalize/standardize the resource consumption. For example the CPU utilization can be converted to cycles/second and n/w utilization can be converted to actual bits/sec. These converted values can now be compared in a meaningful fashion.

3.3 Constraints

The constraints to be satisfied are divided into 2 groups i) virtual machine specific and ii) physical machine specific. With each VM_i is associated a list of M_i constraints $\mathcal{VC} = \{VC_{(1,i)}, VC_{(2,i)}, \ldots, VC_{(M_i,i)}\}$. Similarly, L_i constraints are associated with physical machine $\mathcal{PC} = \{PC_{(1,i)}, PC_{(2,i)}, \ldots, PC_{(L_i,i)}\}$. As mentioned in Section 2, these constraints can originate at system, SLA, legal and application level. Typically application, SLA and legal constraints are associated with VMs while system constraints are associated with physical machines. For example, if due to legal policies or client commitments, two applications cannot reside on the same machine, then such exclusivity constraints are forced on the VMs for both the applications. However, if a physical machine has available bandwidth of 100 Mbps then such constraints are enforced at physical machine level to check if it can cater to the network need of VMs which can be potentially hosted onto the machine. *Whenever unambiguous, we denote a constraint by C ignoring the prefix P or V.*

We also extend the traditional definition of a constraint to include time component, $C_{(j,i)}^{[t_1,t_2]}$ specifies the j^{th} constraint of i^{th} VM which should hold in the interval $[t_1, t_2]$. The constraint $C_{(j,i)}^{[t_1,t_2]}$ is said to be satisfied if $eval(C_{(j,i)}^{[t_1,t_2]}|\mathcal{P})$ is 1, where $\mathcal{P}$ denotes the properties of the environment/architecture in the time interval $[t_1, t_2]$. and *eval* evaluates the inequality specified in the constraint. For example, in the above mentioned network bandwidth example, the constraint is of the form $\forall_{t=1}^{10} \sum_{i \in [1,S]} O_{l,i}^t$ < 100 Mbps, where O_l is the sensor monitoring network usage, time period is [1,10] and S is set of potential VMs which can be consolidated and N/W_i is network usage of VM_i. If for a given set of VMs and time period, the inequality is satisfied *eval* returns 1 else 0.

3.4 Optimization Problem Formulation

The problem of selecting n physical machines which can host N VMs $\mathcal{VM}$ is posed as an optimization problem. Such a formulation can also account for fixed and dynamic costs. Initially, each VM (application) is hosted by one physical machine and each physical machine hosts exactly one VM. Therefore, $|VM| = |P| = N$. n is not known apriori and N serves as an upper bound on n. Let $\mathcal{A}$ be a $N \times N$ matrix such that $A_{i,j}$ =1 specifies that VM_i is assigned to P_j. Y denotes a N bit vector, such that $Y_i = 1$ implies that P_i is current being used and one or more VM_j's are assigned to it. At start of the analysis A is a diagonal matrix. Similarly, P is a vector of all 1's. $Cost_i$ specifies the fixed cost incurred in if P_i is active. $MCost_{i,j}$ is the cost for migrating VM_i to P_j. Typically, migration cost within the same cluster is cheaper than inter cluster ones. We use the VMs to application mapping specified in $\mathcal{H}$ to analyze this aspect in detail in Section 4. $\mathcal{F}$ calculate the dynamic cost of a physical machine if one or more VMs are assigned to it. Currently, the function uses the CPU utilization for power computation. Finally $[t_1, t_2]$ is the consolidation window. The total benefit attained by packing the VMs can be calculated by the function in Equation 1.

The valid assignment can be obtained by maximizing the objective function. The first term, i.e., $\sum_{i=1}^{N}(Cost_i + \mathcal{F}(CPU_i))$ accounts for the initial fixed as well as the power cost. However, this term is fixed and does not change while maximizing the function. Therefore, for computational efficiency we can ignore the first term. Now the objective function is:

$$B = \sum_{j=1}^{N}(Cost_j \times Y_j + \sum_{k=1}^{N} MCost_{k,j} \times A_{k,j} + \mathcal{F}(\sum_{k=1}^{N} CPU_k \times A_{k,j})) \quad (2)$$

The function now calculates the cost of re-consolidation and a valid assignment will result when the objective function is minimized subject to the following constraints.

Constraint 1: Each VM_i is assigned to exactly one P_k, i.e., each column in A should add upto 1

$$\forall i \in [1, N], \sum_{j=1}^{n} A_{i,j} = 1 \tag{3}$$

Constraint 2: Every VM_i should be assigned to some P_k, i.e., A should have exactly N non-zero entries.

$$\sum_{i=1}^{N} \sum_{j=1}^{n} A_{i,j} = N \tag{4}$$

Constraint 3: All VM specific constraints are satisfied.

$$\forall_{i=1}^{N} \forall_{j=1}^{M_i} eval(VC_{i,j}^{[t_1,t_2]}) = 1 \tag{5}$$

Constraint 4: All physical machine specific constraints are satisfied.

$$\forall_{i=1}^{N} \forall_{j=1}^{j=L_i} eval(PC_{i,j}^{[t_1,t_2]}) = 1 \tag{6}$$

Constraint 5: Finally, if some VM_i is assigned to P_j then Y_j should be 1

$$A_{i,j} \leq Y_j \tag{7}$$

3.5 Dynamic Reconsolidation

Assume the consolidation window size is T_s. First we minimize the optimization function in interval $[1, T_s]$ and generate assignment matrix $A_{[1,T_s]}$. While consolidating for next time span $[T_s + 1, 2 * T_s]$, we use the set of new constraints (valid in the time interval) and use $A_{[1,T_s]}$ as the starting point for the optimization routine. This intelligent initial value assignment provides computational efficiency to our framework because for the VMs in which there is no significant change in behavior (resource consumption, constraints etc), we already have a valid assignment. Only minor modifications are needed for such VMs and the algorithms converges very quickly.

3.6 Static Reconsolidation

Generating static reconsolidation recommendations is simply a special case in our framework. All the migration costs are set to zero,i.e., $\forall_{i=1}^{N}$ $\forall_{j=1}^{N}$ MCost(i,j)= 0. The consolidation window is set to cover the whole time period, i.e., t_1 =1 and t_2 =T. With these settings the algorithm reduces to static case.

4 Experimental Validation

We applied *ReCon* on trace data collected from an actual production environment. The trace data was collected using a monitoring system we built called MDMS (Model Driven Monitoring System) [8]. MDMS is deployed in two production environments and provides a rich source of data from hundreds of servers to explore the applicability of server consolidation using virtualization in these environments. In a particular environment where MDMS monitors over 1500 non-virtualized servers divided into various clusters, we took a sample set of clusters to apply the *ReCon* tool and generate consolidation recommendations. In this section we present various experiments performed using the tool highlighting its effectiveness and usefulness. But first we describe the dataset used and the associated setup.

4.1 Dataset

We used the monitored data collected from 186 physical servers located in a large data center. The identity and the industry sector of the organization that is supported by the data center has been kept anonymous for privacy reasons. The servers are divided into 35 clusters with each cluster supporting one application, for e.g. *Billing*, *CRM* etc. A server does not run more that one application and thus does not belong to more than cluster. Currently approximately 15 parameters are monitored for every server on the average. These parameters include CPU utilization, wait I/O, network utilization and more. In this paper we assume independence among the monitored parameters and only use the CPU utilization parameter to capture the resource requirement of the virtual machine.

Cost	Total Recommendations	Inter-cluster	Intra-cluster
40%	2543	150 (5.8%)	2393(94.1%)
50%	2352	181 (7.7%)	2171 (92.30%)
60%	1921	219 (11.4%)	1702 (88.6%)
70%	1556	329 (21.14%)	1227 (78.86%)

Table 1: Results to show how the change in migration cost affects the number of recommendations.

In Section 6, we briefly point to the modifications which can enable the proposed algorithm handle multiple resource utilization parameters. The parameters are sampled at 5 minutes interval and sample dataset for 186 servers is taken for a period of one month resulting in $8,928$ samples per server and $1,660,608$ data samples over all servers. The optimization model is specified in AMPL [2] and CPLEX [7] is used as solver. *ReCon* is run on on a machine with 2.13 Ghz Intel processor and 1 GB main memory.

4.2 Evaluation Metrics

The evaluation of *ReCon* is captured in it's time efficiency or how fast it works given the size of the data. This is important from a tooling perspective because it is difficult to perform any what-if analysis if the tool takes a long time to generate reports. The efficiency is measured by comparing the consolidation window size (T_S) to the time taken by *ReCon* (T_R) given the number of source and target servers. The efficiency $\mathcal{E}$ is given by $\frac{T_R}{T_S}$. For an highly efficient algorithm $\mathcal{E} \sim 0$. For example, if *ReCon* recommendations are generated in 2 minutes with a 1 hour trace input for 100 source and target servers, then $\mathcal{E} = \frac{2}{60} = .03$. The value of $\mathcal{E} \geq 1$ renders the algorithm useless for all practical purposes the analysis takes more time than the time span of the collected data. *Essentially, $\mathcal{E}$ measures the rate at which data can be analyzed and (new) recommendations can be generated.*

The effectiveness of *ReCon* applied on a trace is measured by savings in terms of percentage of physical machines that can be turned off by packing N VMs onto n_i ($N \geq n_i$) physical machines while satisfying all the constraints in the i_{th} consolidation window. Please recall, S is size of consolidation window, T represents the entire trace period and $\frac{T}{S}$ is total number of windows and therefore the number of times recommendations are obtained in a dynamic setting. The savings within a i^{th} window is $\mathcal{S}$ is given by $\frac{N-n_i}{N}$. The total saving over all windows is given by $\sum_{j=1}^{\frac{T}{S}} \frac{N-n_j}{N}$. The higher the value of $\mathcal{S}$ the higher the savings. $\mathcal{S} \sim 1$ implies most of the physical machines can be turned off resulting in huge savings. This measure is heavily dependent on the trace. The intent of our experiments on the trace is to illustrate what kind of micro-measurements are generated in a report rather than the results themselves. The merging of two VMs onto the same physical server is called a "recommendation" in this section. We also assume that the dynamic cost of a server varies linearly with utilization in all experiments. We can experiment with higher degree models in the tool to stress the optimization backend.

4.3 Change in recommendations vs migration cost

In this section, we study how the recommendations vary as the migration cost is varied. Migration cost is the key element of dynamic consolidation. The inter cluster migration cost is normalized to be 100 whereas the cost of intra-cluster migration is varied (as percentage of inter cluster migration cost). Table 1 enumerates the number of recommendations generated and partitions it into inter and intra cluster recommendations. There are two key observations to be noted from these results.
Observation 1: It is evident that the number of recommendations decreases as the intra-cluster cost increases. High migrations costs will not allow VMs to be packed together unless the CPU utilization of each VM is very small. In such cases the fixed cost might exceed migration cost and new dynamic server cost, thereby allowing the reconsolidation. However, if the CPU utilization is very high, then the resultant dynamic server cost will also be very high, which in turn will result in negative profit (loss). Therefore such recommendations will not be generated.
Observation 2: The gap between percentages of inter-cluster and the intra-cluster recommendation decreases as the intra cluster migration cost approaches inter-cluster migration cost. This is expected because the cost of an inter-cluster reconsolidation matches with intra-cluster reconsolidation.

Cost	Total Recommendations	Peak-Trough	Low Utilization
40%	2543	190 (7.4%)	2353(92.5%)
50%	2352	174 (7.3%)	2178 (92.6%)
60%	1921	139 (7.2%)	1782 (92.8%)
70%	1556	121 (7.7%)	1435 (92.2%)

Table 2: Results showing the change in distribution of recommendations with varying migration cost

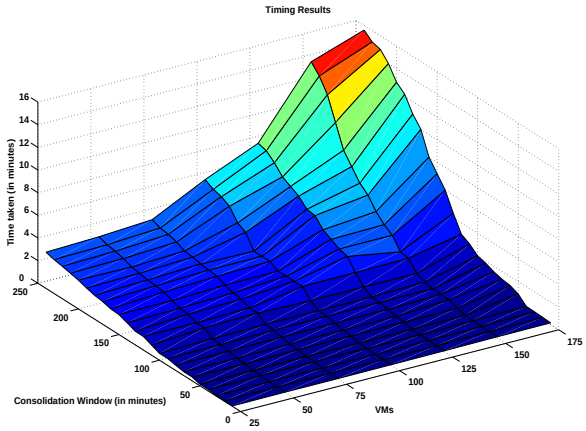


Figure 5: Surface plot showing the timing results as consolidation window and number of VM are varied.

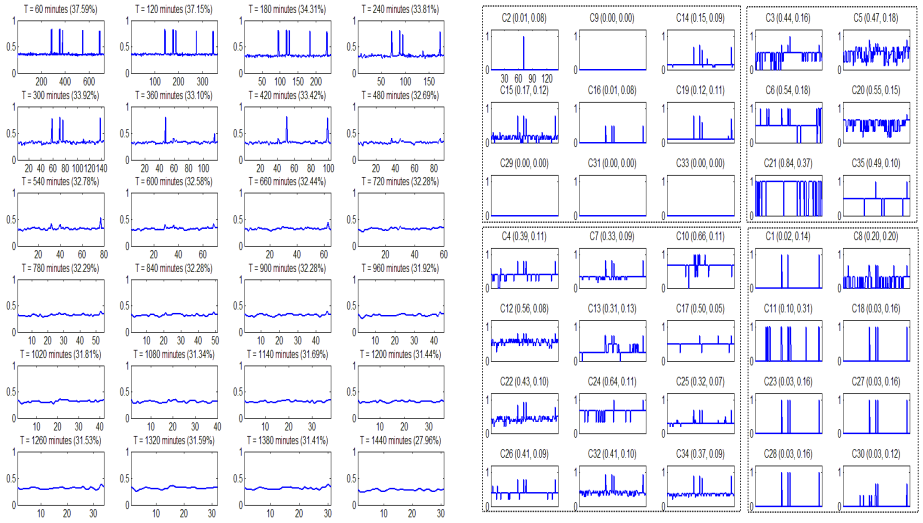
4.4 Cause of recommendations

As mentioned in Section 1 our tool framework can naturally discover pairs of servers which either have low utilization or display a peak-trough kind of pattern. These pairs are then recommended for consolidation. In this micro-measurement we aim to find the number of recommendations in each category. Assume A and B are two VMs recommended to be consolidated with M_A and M_B as monitored values and $|M_A| = |M_B| = N$. The series M_A is transformed into a digital series D_A: $\forall i \in [1, N]$, $D_A(i)$ =1 if $M_A(i) \geq \mu(M_A)$, 0 otherwise. The recommendation is considered to be arising from peak-trough(PT) pattern if $PT(A, B) \geq N * \epsilon$, where $\epsilon \in [0, 1]$ and $PT(A, B)$ is given as: $PT(A, B) = |XOR(D_A, D_B)|$. Essentially PT calculates how many times the signals show complimentary behavior. An example case in shown in figure 3(a) whereas Figure 3(b) shows an example where VMs are consolidated because of low CPU utilization. Table 2 documents the recommendations with the cause. It is evident that the number of recommendations due to peak-trough pattern is much smaller than the low utilization ones. However, this measurement shows that our approach can capture these without checking explicitly for peak-trough patterns.

4.5 Efficiency results

Figure 5 shows the effect of changing number of VMs and consolidation window on the time taken by the optimization method. The number of VMs to be analyzed are varied from 1 to 175 (in interval of 25). The consolidation window is varied between 10 minutes to 240 minutes (in interval of 10 minutes). While varying both the parameters, timing results were obtained for 168 runs. To decrease the effect of the actual trace values, we ran the experiment 4 times with different parts of the trace. Finally, timing results from different runs were averaged to produce the surface plot.

As the number of VMs increases, the time taken increases in a linear fashion. This increase is expected because with increase in VMs, the number of constraints also increase and therefore, the optimization model takes more time. Increase in the time as the consolidation window size is increased is due to more data points and is self-explanatory. Finally, when 240 minutes is used as consolidation window size and 175 VMs are used, the time taken by our algorithm is only 15 minutes. Presence of such a property facilitates performing the analysis at run time (c.f. Section 7).



(a) (b)

Figure 6: Experiments capturing different aspects in a dynamic setting. The y-axis in all plots the % of machines which can be turned off (a) shows change in savings over a month at different consolidation window sizes. (b) shows the savings over a month for each cluster. The consolidation window size is 300 minutes.

4.6 Change in recommendations over time period

In this experiment, we study how the recommendations vary with change in the consolidation window size. Figure 6(a) documents these change for 24 different window sizes over a period of one month. The x-axis in each subplot shows number of consolidation windows, i.e., (total minutes in month)/(window size). The y axis enumerates the total savings. The first subplot shows shows savings when consolidation is perform at an hourly basis whereas the last subplot shows the saving at an daily basis. The number in parenthesis on each plot shows total percentage of servers which can be turned off. The key points which can be inferred from the figure are:

Observation 1: As the time window is increased the number of recommendations (and thereby the savings) decreases. For example at window size of 60 minutes, the 37% of servers can be switched off whereas at 1440 minutes this saving gets reduced to 27%. This is expected because increase in window size imply more samples which in turn makes it difficult to satisfy the constraints. Essentially at smaller window size, we have to satisfy *local* constraints however at larger window constraints become *global*. For example, consider a toy example where A and B are two VMs. A_{cpu} = { 10, 10, 30, 80, 70, 60} and B_{cpu} ={30, 40, 80, 30, 20, 30} are 6 consecutive CPU utilization samples synchronized in time for the two VMs. Assume the constraint that the summation of the two CPU utilization sequences can go over 100% only once in the chosen time span T_s (equal to 6 samples). If A and B are hosted on same machine, the resulting machined CPU utilization will be {40, 50, 110, 110, 90, 70}. Now it is clear than constraint is not violated if $T_s \leq 3$, the combined CPU goes over 100% exactly once, however at $T_s > 3$, the constraint is not satisfied and hence the packing is not a valid one.

Observation 2: This experiment can be used to find the time span which is small enough to capture the dynamic behavior and at the same time it should be large enough that the optimization engine is not used repeatedly without any gain. In this example $T = 300$ minutes seems like a good choice for the time period.

4.7 Change in clusters over time

In this experiment, we study the effect of recommendations on the individual clusters. The plots for all 35 clusters are shown in Figure 6(b). The y-axis shows the percentage of machines which can be turned off, if recommendations are taken into account. The time span T_s is chosen as 300 minutes. The x-axis range from 1 to 144, total 300 minute periods in a month (43200 minutes). The mean and standard deviation of savings for each cluster are shown in the parenthesis. Based on the mean savings and the variation in savings, the clusters can be divided into four groups:

Low Variation - Low Saving: This set represents clusters which consistently provide low savings.

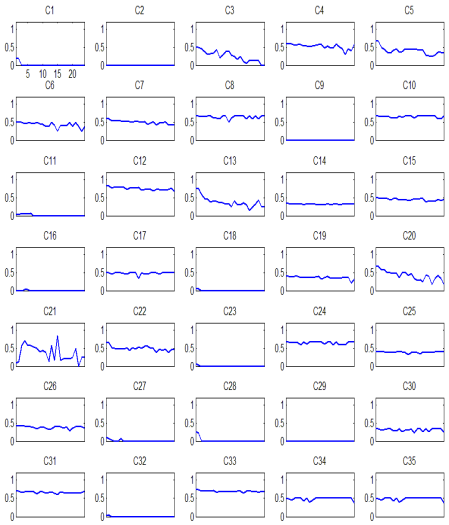


Figure 7: Experiments capturing different aspects in a dynamic setting. The y-axis in all sub-plots the % of machines which can be turned off shows savings for each cluster as consolidation size is varied from 1 hour to 24 hours (x-axis).

These clusters offer no savings throughout the month. Based on this analysis and other domain specific knowledge, the admin may choose to ignore these clusters next time ReCon is used. Clusters in this set are shown in top-left quadrant in Figure 6(b).

Low Variation - High Saving: This group represents most profitable clusters with very low overhead (in terms of migration and switching machines on/off). The profile of such clusters do not change considerably over time. Bottom left quadrant shows these clusters.

High Variation - Low Saving: These clusters typically offer very little savings. Please note that in some time interval, the savings are very high. However, in other all the machines in these clusters are used. Clusters in top right quadrant highlight this behavior.

High Variation - High Saving: Clusters in these groups have very dynamic profile. Bottom right quadrant shows the clusters in this group. ReCon exploits the dynamic profiles of such clusters to maximize the profit.

Figure 7(c) shows the savings for each cluster as the consolidation window size is varied. Such a plot can be used to further analyze each cluster in more detail. For example, clusters 1 and 28 show small savings at very low window size. Similarly, clusters 13 and 20 show decreasing trend whereas clusters 8, 14 and 15 show constant trends. This allows one to select the right consolidation window size for the whole setup. It is also possible to partition the clusters into groups and use a consolidation window suitable for specific groups of clusters. Thus inter-cluster migrations can only happen across clusters with the same consolidation window.

5 Related Work

In this section we describe some of the related work in dynamic VM sizing and consolidation planning. There is significant work in capacity planning [10, 6] and runtime resource management domain without bringing in the aspect of virtualization. Recently there have been products and research papers in this area in the context of virtualized platforms. VMWare's Distributed Resource Scheduler (DRS) is a runtime component that monitors the utilization of physical servers and virtual machines and allocates resources to the virtual machines based on various policies [15]. When resources are constrained for a VM on a physical server then they may be also migrated to a different server using VMotion [14] migration technology.

Bobroff et al. [3] describe algorithms for reconsolidation in a dynamic setting while managing SLA violations. They have developed an analytical formula to discover VMs which will benefit the most from dynamic migration. A bin-packing heuristic is then used to pack multiple VMs onto a physical server. *ReCon* has different goals as it is architected as a trace-driven planning tool with reporting capabilities.

In the static consolidation area, several bin-packing heuristics [12, 9] have been used to map VMs to physical machines. Typically consolidation tools tackle the problem using static benchmark

numbers for servers to consolidate larger number of physical servers as VMs onto a large server. Even if actual measurement data is used from an existing environment, the algorithms are fed with aggregate resource utilization numbers to find out a static VM to physical server mapping.

ReCon uses measurement data and allows the user to divide the trace into consolidation windows and create a dynamic mapping between VMs and physical servers. It allows the user to play with different parameters on the same trace and generates micro-measurement reports on dynamic consolidation plans. The target is keep the core tool platform independent with specific models plugged in for different target environments. The IBM Server Planning Tool (SPT) [13] specific to the Power architecture is the closest match to *ReCon*. It however takes a static consolidation planning approach. Some of the algorithms and what-if analysis capabilities in *ReCon* can be incorporated in SPT. VMWare also has a tool and a professional service offering to plan the long term virtualization strategy of a data center [11].

6 Discussion and Future Work

Currently, we are investigating the extensions to the proposed framework in the following directions.
Handling Multiple Attributes: Even though we used only CPU utilization as the resource attribute, the framework is capable of handling multiple attributes and associated constraints. The constraints on all attributes should be expressed in the form of (in)equalities and there should be associated *eval* functions. No change in required to the core optimization engine. However, in the current implementation our algorithm still cannot exploit relationships or correlation among the attributes. For example, there may be a time lag between high CPU utilization and high I/O utilization. Without considering the time lags, the obtained savings will be less than the optimal savings. Accounting for the interdependence among the attributes forms major part of our future initiatives.
Run Time Reconfiguration Tool: In order to convert the planning tool into a real time decision module, we need a highly efficient implementation and forecasting logic. It is clear from the results in Section 4 that the alorithm is very time efficient. We plan to employ machine learning methods to fit a model (or mixture of models) to the historical data [5, 1, 4] and predict future resource consumption based on the model. This forecasted data will then be used to generate recommendations in a real time proactive fashion.
Extending What-if Analysis: Currently, our tool allows the system administrator to change various parameters used in the optimization procedure and study the effect on time taken and savings. However, we believe that much more useful and interesting scenarios can be handled. Please recall, the physical machine level constraints taken into account on the specification of the machine. The system administrator can change these constraints to hypothetically change the available resources and evaluate the savings. For example, it can be discovered that by increasing the n/w bandwidth of one specific physical machine one can migrate more VMs to this machine and obtain increased savings. Similar scenarios can be explored by varying processor speeds etc.

7 Conclusion

In this paper, we presented a planning tool called *ReCon* to recommend application consolidations in a large multi-cluster data center. The tool analyzes the resource consumption data of various applications, discovers applications which can be consolidated and subsequently generates the static or dynamic consolidation recommendations. We formulate the problem in an optimization framework wherein valid recommendations are generated while minimizing the total number of physical servers and satisfying the constraints. The optimization framework provides the flexibility to specify and impose *system, application* and *legal* level constraints. Moreover, time varying constraints are easily incorporated to account for temporal change in workload characteristics. Finally, different migration cost functions, virtualization models and server cost models can be plugged into the tool. Overall, we believe that our methods provide the flexibility to the system administrator to specify most of the real life considerations while deriving the recommendations.

We performed experiments on a dataset collected from a large production environment. On an average, *ReCon* was able reduce the number of servers by 40% based on the specific trace. It was also evident from the timing results that the core algorithms are efficient enough to be adapted for a run time module.

References

[1] S. Akioka and Y. Muraoka. Extended forecast of cpu and network load on computational grid. In *CCGRID*, pages 765–772, 2004.

[2] AMPL. `http://www.ampl.org`.

[3] N. Bobroff, A. Kochut, and K. Beaty. Dynamic placement of virtual machines for managing sla violations. In *Integrated Network Management*, 2007.

[4] G. Box, G. Jenkins, and G. Reinsel. *Time Series Analysis: Forecasting and Control*. Prentice Hall, 1994.

[5] P. Brockwell and R. Davis. *Introduction to Time Series and Forecasting*. Springer Verlag, New York, 2002.

[6] J. S. Chase, D. C. Anderson, P. N. Thakar, A. Vahdat, and R. P. Doyle. Managing energy and server resources in hosting centres. In *SOSP*, 2001.

[7] CPLEX. `http://www.ilog.org/cplex`.

[8] B. Krishnamurthy, A. Neogi, B. Sengupta, and R. Singh. Data tagging architecture for system monitoring in dynamic environments. In *IBM Research Report– RI070XX, Submitted to NOMS'08*, 2007.

[9] W. Leinberger, G. Karypis, and V. Kumar. Multi-capacity bin packing algorithms with applications to job scheduling under multiple constraints. In *ICPP*, 1999.

[10] Z. Li, C. H. Xia, P. Momcilovic, and L. Zhang. Ambience: Automatic model building using inference. In *Congress MSR03*, 2003.

[11] VMWARE Planner. `http://www.vmware.com/products/capacity_planner/`.

[12] J. Shahabuddin, A. Chrungoo, V. Gupta, S. Juneja, S. Kapoor, and A. Kumar. Stream-packing: Resource allocation in web server farms with a qos guarantee. In *HiPC*, 2001.

[13] Server Planning Tool. `http://www-304.ibm.com/jct01004c/systems/support/tools/systemplanningtool/`.

[14] VMotion. `http://www.vmware.com/products/vi/vc/vmotion.html`.

[15] VMWARE. `http://www.vmware.com/products/vi/vc/drs.html`.