BMC Software, Inc.

Technical Disclosure Publication Document

Discovering correlations between server configuration and power consumption to enable power savings

Neeran Karnik

Abhay Ghaisas

Overview

This document describes a way of potentially reducing power consumption in a data center by means of discovering correlations between server configuration and power consumption.

Background

The basic problem addressed is the increasing power consumption in datacenters, and the need to provide tools that help control it. Beyond coarse-grained approaches such as virtualization, that reduce the number of physical servers needed, there is a need to study how power consumption varies with various server configuration parameters. There are, however, so many different parameters, with complex interplay among them, that it is very difficult to build accurate and useful models a priori that let the customers control and reduce power consumption. This invention solves this problem.

Solution

In a sample application of this solution, there is a correlation between server and application configuration and power consumption. These correlations are discovered using a data-mining technique known as Association Rule Mining (ARM). For example, a certain swapsize, or virtual memory setting may result in higher power consumption (because of more frequent usage of the disk). Systems with solid-state disk caches have lower power consumption than those without. A correlation between a RAID level and the corresponding power consumption can be made. There are many such possibilities, and the ARM technique is capable of discovering them statistically, without any in-built a priori knowledge of the semantics of the parameters.

For achieving this, first of all, power consumption by a server must be measured. This is accomplished by deploying a small piece of software on the target server (called agent). This agent will have to interact with platform-specific Application Programming Interfaces (APIs) for this purpose. As an example, IBM's Power6 and Power7 chip-based Unix servers have the

capability of reporting cumulative power consumption data. If remotely accessible APIs are made available by a platform, this data could be accessed without using an agent.

Periodic snapshots of such servers can be taken, which capture this power data along with a set of server parameters that may potentially have an impact on power consumption. These server parameters may include configuration parameters that change relatively infrequently as well as parameters that indicate the current state, e.g., CPU & memory utilization, or even specific lists of running processes, that change more regularly. The periodicity of snapshots can be decided based on how frequently we expect these parameters to change.

By taking periodic snapshots of a number of servers in the datacenter, we build up a significant corpus of data in a logical table. Each row in such a logical table corresponds to one snapshot of a server configuration and state, including the power consumption between consecutive snapshots. Note that the power consumption recorded in not cumulative. We now subject this table to data mining techniques, such as association rule mining (ARM).

ARM can discover rules that correlate attributes in the table. These rules take the form

$$(x, y, z) \Rightarrow (a, b)$$

This means that when a particular variable X takes the value "x," another particular variable Y takes the value "y," and another particular variable Z takes the value "z," it is statistically likely that some other variable A takes the value "a" and that another variable B takes the value "b." In its usual form, ARM is not aware of the real world semantics of any of these variables, and it attempts to discover any and all such correlations that exist in the data. However in this example case the interest is in a right-hand side that says "Power consumed=High" or "Power consumed=Low", etc. The ARM algorithm can be constrained to generate only those rules with the power consumption attribute on the right-hand side. These rules can be surfaced to the user, who will thus discover that a certain combination of configuration parameters leads to high (or low) power consumption. This knowledge can then be used to tune the applications/servers in the datacenter, so as to reduce the overall power usage in the datacenter. The settings can be documented as best practices used in reference configurations, and rolled out to other servers that have different values for the configuration by using some automated server configuration management tools.

The ARM algorithm can be tuned using two statistical parameters – support and confidence. We can provide default values for these parameters, but allow the user to tweak them and regenerate the rules using the algorithm. If these knobs are loose, ARM can generate large numbers of rules, in the process potentially discovering rules that it might otherwise have missed. The user, however, can be overwhelmed by the large number of rules. A tighter setting of these knobs can reduce the number of rules generated, making it easier for the user to study the results – the tradeoff being that some potentially inherent correlations might be missed.

A further modification may be necessary to ensure that this method delivers value in a highly virtualized environment. Power consumption data is only available from physical servers, each of which may host several virtual guests. In this scenario, some application-level configuration

parameters can be collected from the virtual guests, some virtualization-specific parameters (e.g. number of virtual machines (VMs) hosted, their CPU/memory allocations etc.) added, and these parameters may be 'collapsed' – possibly involving aggregation functions such as averaging – into the table row representing the physical server. This aggregation can be weighted by the amount of time that a guest VM was active (and thus consuming its share of power).

In other words, the set of attributes of a physical server are expanded to capture virtualization information as well as application data from its guests. The subsequent ARM analysis is then able to identify the inherent rules or correlations among this expanded set of attributes.