Energy Aware Consolidation for Dynamic Resource Applications
May 6th, 2015

Adrian J. Mirabel and Rashid Siddiqui
Overview

- Introduction
- Original Paper Review
- Method Description
- Dynamic Allocation Algorithm
- Method Example
- Simulation
- Analysis of Method
- References
- Questions
The goal is to find an method for allocating client applications on a set of common servers in a manner that minimizes energy used.

The method proposed is an extension of the one described in the original paper *Energy Aware Consolidation in Cloud Computing*

The original method is limited to allocating applications with static resource utilizations.

Our extended method is capable of allocating applications with dynamic resource footprint.
What is Energy Aware Consolidation?

- Running many dissimilar client applications on the same server cluster.
- In other words running multiple data center applications on a common set of servers.
- This allows for the consolidation of application workloads on a smaller number of servers that may be kept better utilized.
Effective consolidation is not as trivial as packing the maximum workload in the smallest number of servers.

- Keeping resources at 100% utilization is not energy efficient.
- Goal is to minimize the energy used per unit service.
- Energy optimal resource point (vector) is 50% hard disk, and 70% cpu utilization.
The allocation method is almost identical to the method proposed in the original paper, except for the *

1. For each server type, the server is subjected to various client application workloads in order to obtain Energy consumption data. From this data we obtain:
   a. the Energy vs. resource utilization relationship for the server(s).
   b. the Energy optimal resource point. ex) [70%, 50%]
   c. * the Energy vs. resource usage relationship for the applications.
      i. either tabulated data or function.

2. * Allocate incoming client applications according to the Dynamic Allocation Algorithm.
Resource utilization and consumption by servers and client applications are conveyed with resource vectors.

- ex) 50% CPU utilization and 50% Hard Disk utilization would be written \((0.5,0.5) = r\)

Distance between two resource points is given by the euclidean distance \(\delta_e = \sqrt{x_1^2 + x_2^2 + .. + x_n^2}\)

- ex) \(r_1 = (0.3,0.8), r_2 = (0.7,0.5)\), \(\delta_e(r_2 - r_1) = \sqrt{(0.7 - 0.3)^2 + (0.5 - 0.8)^2} = 0.5\)

\(r(t)\) indicates a resource vector is dependent on time.

\(\text{max}\{r(t)\}\) returns a set of resource vector points that are local maxima.

\(s^*=(0.7,0.5) = \text{optimal resource vector for server s.}\)

\(|. . |\), is the cardinality of set, it give the number of elements in the set.
Consider a workload $W$, and a set of servers $S$ the allocated the algorithms as follows.

**Dynamic Allocation Algorithm**

1. Let score[$s$] be a map of mean distances of maxima for a server.
2. Foreach client application $w$ in $W$
   a. Foreach server $s$ in $S$
      i. Compute $\text{max}\{s\} \cup \text{max}\{w\} = \text{max\_set}$
      ii. Foreach $t$ in max\_set
         1. IF $r_s(t) + r_w(t) \leq s^*$ THEN score[$s$] += $\delta_e(s^* - (r_s(t) + r_w(t)))$
         2. ELSE remove $s$ from score and break.
      iii. $\text{score}[s] /|s|$
   b. Assign $w$ to the maximum scoring server.
We have a workload consisting of two applications with resource utilizations of the following form:

- App 1 has a resource relationship given on the right.
  - The app has a maxima at $t = 16$ and resource utilization of $r_1$
- App 2 has a resource relationship given on the right.
  - The app has a maxima at $t = 31$ and resource utilization of $r_2$

The graphs are for both CPU and HD utilization vs. time for each application.
Assume we have two servers, A and B that are idling at 10% utilization, so $s_a = (0.1, 0.1)$, $s_b = (0.2, 0.2)$.

- Both servers has optimal resource $s^*(t) = (0.7, 0.5)$

First we compute the max_set for $\max\{s_a\} \cup \max\{r_1\} = \{t=16\}$

Now we check if the $r_1(t=16) + s_a(t=16) < s^*(t)$, $(0.2, 0.2) + (0.1, 0.1) < (0.7, 0.5)$

Next we add the euclidean distance between the target

- $\delta_e(s_a(t) + r_1(t), s^*(t)) = \sqrt{(0.3 - 0.7)^2 + (0.3 - 0.5)^2} = 0.447 \rightarrow \text{score}[s_a]$
- $\delta_e(s_b(t) + r_1(t), s^*(t)) = \sqrt{(0.4 - 0.7)^2 + (0.4 - 0.5)^2} = 0.316 \rightarrow \text{score}[s_b]$

Now we assign the application to the server with largest score, which is $s_a$. 
Assume we have two servers, A and B that are idling at 10% utilization, so $s_a = (0.1, 0.1), s_b = (0.2, 0.2)$.

- Both servers have optimal resource $s^*(t) = (0.7, 0.5)$

First we compute the max_set for $\max\{s_a\} \cup \max\{r_1\} = \{t=16\}$

Now we check if the $r_1(t=16) + s_a(t=16) < s^*(t), (0.2, 0.2) + (0.1, 0.1) < (0.7, 0.5)$

Next we add the euclidean distance between the target

- $\delta_e(s_a(t) + r_1(t), s^*(t)) = \sqrt{(0.3 - 0.7)^2 + (0.3 - 0.5)^2} = 0.447 \rightarrow \text{score}[s_a]$
- $\delta_e(s_b(t) + r_1(t), s^*(t)) = \sqrt{(0.4 - 0.7)^2 + (0.4 - 0.5)^2} = 0.316 \rightarrow \text{score}[s_b]$

Now we assign the application to the server with the largest score, which is $s_a$. 
In order to validate the algorithm, a simulation of the original experimental conditions was constructed.

The simulation consists of four servers processing a workload of 8 client applications.

The Power vs. resource utilization for the simulated servers was obtained from the Energy per transaction graph shown below.
We want **Power** as a function of cpu and hard disk utilization.

The original experiment gives data for the total energy used over a 60-second period for constant utilizations.

We use the thermodynamic relationship **Power** * Time = **Work**. and make the approximation that **Power** is dependent only on the utilization rates. This approximation is only true for small time intervals.

- c = CPU utilization rate, and h = Hard Disk utilization rate.

E(c,h) / N is given, and we know that P(c,h) * 60 = E(c,h)/N. Solving for P(c,h) yields P(c,h) = E(c,h) / (N * 60).

The **Power** as a function of resource utilizations is critical for the simulation, since P(c,h) * dt = Energy used for the small time interval dt.
Assumption:
- Profiling data has already been obtained for all client applications.
- All servers are homogenous with identical Power vs. resource relationship.
- All servers are awake.

Simulation Description:
- Processing a workload consisting of dummy applications with variable resource utilizations.
- At simulation start, the allocation algorithm begins delegating applications to each of the servers.
- Server processing takes place in discrete time steps ~0.1 (ms).
  - The energy used during each timestep is calculated using $P(c,h) \times$ (timestep), and added to the total.
- The simulation concludes when the workload has been processed.
The simulation was run for three allocation algorithms using a workloads of varying utilization functions.

- Dynamic algorithm - algorithm for allocating applications with dynamic resource utilizations.
- Original Algorithm - algorithm for allocating applications with static resource utilizations.
- Optimal Algorithm - algorithm that allocates applications according to a apriori calculated ordering that is optimal. This is algorithm is used for validation only.
■ For a workload mix with a constant utilization of 180% CPU and 180% HD.
  ○ Original Algorithm ~ 4.12 [J/Op]
  ○ Dynamic Algorithm ~ 4.12 [J/Op]
  ○ Optimal Algorithm ~ 4.1 [J/Op]

■ For a workload with mix of constant and sinusoidal utilizations.
  ○ Original Algorithm ~ 6.18 [J/Op]
  ○ Dynamic Algorithm ~ 4.5 [J/Op]
Questions?