閱讀摘錄與筆記

Dynamic Resource Allocation Using Virtual Machines for Cloud Computing Environment (2013)

Load balancing system:

- to allocate data center resources dynamically based on application demands (overload avoidance)
- support green computing by optimizing the number of servers in use (number of used Physical Machine minimized)

這兩個目標是相抵觸的。想要避免 overload 的話就要我們會希望各機器的使用率被分散得平均且低。想要 green computing （節能）的話就希望讓啟動的機器越少越好，就會增加啟動中機器的使用率。

本來以為沒什麼屁用，因為他其實要做的事情就是：

- $MIN_{啟動的PM}(MAX\ 不會overload的可能性)$

- $MIN_{啟動的PM}(MIN\ overload的可能性)$

如果是以傳統的 NASA 管理者來說，這件事情其實可以被計算並管理好的。

但是以 data center 的角度來說，
使用者使用的情況有限的情況下（不知道裡面的實際應用情況）要去自動化這件事情可以**維護 data center 的 availbaility** 。

"Skewness" to measure the unevenness in the resource utilization of a server.

Contribution:

- overload avoidance + green computing system
- "skewness" as measure
- load prediction model

Structure:

- bunch of Physical Machine with Xen (1 PM = 1 node)
- Usher as VM control framework (modulized calls for flexibility, 於是這篇論文就是在 usher 上做一個 plugin 來讓 usher 來控制底下 PM 們的 utilization)

Local Node Manager (LNM, at domain 0) 監控 VM: (以 node 為單位，再送給 Usher Central Control)

- CPU and network usage can be calculated by monitoring the scheduling events in Xen
- Memory cannot be observed, solved by self-implementing **working set prober** (WS Prober) to estimate the working set sizes of VM (using random page sampling technique). (1 WS Prober per node)

論文中做的 VM scheduler：

- hotspot solver：超過 hot threshold 就 migrate
- coldspot solver：低於 cold thershold 就 migrate (在低於 green-computing threshold 時才啟用這個判斷)

Load prediction: (based on past external behavior -- collected by LNM)

用 Exponential Weighted Moving Average （指數均線）

但是均線是有延遲性的，而根據資料均線會保守的估計而不是 over-estimate 。

一般的 EWMA

$$E(t) = \alpha \times E(t-1) + (1-\alpha) \times O(t), 0 \leq \alpha \leq 1$$

- $E(t)$: expected load
- $O(t)$: observed load

這篇論文引以為傲的「innovative approach」是把 α 改成負數。

$$E(t) = -|\alpha| \times E(t-1) + (1+|\alpha|) \times O(t), -1 \leq \alpha \leq 0$$

- $\uparrow \alpha$ for increasing trend
- $\downarrow \alpha$ for descending trend

$$= O(t) + |\alpha|(O(t) - E(t-1))$$

並且在上升以及下降時分別用不一樣的 α 來預測。這樣子對上升可以更加高估可能的使用量，對下降也可以更保守的估計。（論文中也提到其他 work 中也有用其它的 prediction algorithm ：linear autoregression (AR)）

Prediction model 其實就是最核心需要研究的東西，文中也提到 this needs more future work on load patterns 。

> The prediction algorithm plays an important role in improving the stability and performance of our resource allocation decisions.

定義 Skewness of the resource, for server p and its resources r_i

$$skewness(p) = \sqrt{\Sigma_{i=1}^{n}(r_i/\bar{r} - 1)}$$

- $\bar{r}$: average utilization of all resources
- r_i: utilization of resource i

另外定義一些參數與他們的用法：

- hot threshold：utilization 超過這個就是 hotspot
- green computing threshold：平均 utilization 低於這個總用量時就開始規劃 green computing
- cold threshold：低於這個代表 server 可能只是單純 idle ，就把這個伺服器列入可能關機的對象中
- consolidation limit：限制 active server 中可以被關機的比率

Hotspot migration

對一個伺服器的許多 VM ，找到在 migrate 過後能降低溫度最多的機器（reduce skewness the most）。許多個 server 就會有一個 migration 的 VM list 。

接著挑選要 migrate 到的地方。

> The server must not become a hot spot after accepting this VM. Among all such servers, we select one whose skewness can be reduced the most by accepting this VM. （low utilization 也是 skewed 的一種）

如果對該 VM 找不到 migrate 的地方（ex: 這個 VM 超大，不管搬到哪裡那台機器就會變 hotspot），就在 VM list 中檢查下一個需要 migrate 的機器。

（這樣的 migration 是有可能可以同時挑選多台，作者認為這樣有可能會增加 server loading 因此採取一次 migrate 一台這種比較保守的做法）

Green Computing

Average utilization below green computing threshold.

在關機一台 PM 之前要遷移所有在其上運行的 VM 。搬移的對象必須在搬移後低於 warm threshold 。（避免因為想要嘗試 green computing 而產生 hotspot）

更進一步overdoing ， consolidation limit 限制了 active server 中可以被關機的比率。

Evaluation of algorithm

> Traces are per-minute server resource utilization, such as CPU rate, memory usage, and network traffic statistics.

Traces from:

- Web InfoMall: 中國網頁歷史信息存儲與展示系統
- RealCourse: A Distributed Course Video Shareing System
- AmazingStore: The largest P2P storage system in China

FUSD load prediction: $\uparrow \alpha = -0.2, \downarrow \alpha = 0.7, W = 8$ (default parameter values based on **empirical experience**)

This gives us a chance of future improvement on the load prediciton. There maybe potential concern on the time of training the model. We can adjust of way of learning the model, doing **Incremental Training** we can aggregate our model as historical data increases.