Towards Green Cloud Computing: Demand Allocation and Pricing Policies for Cloud Service Brokerage

Chenxi Qiu
Holcombe Department of Electrical and Computer Engineering
Outline

1. Introduction
2. Algorithm Design
3. Performance Evaluation
4. Conclusion
What is Cloud Service Brokerage

Definition: A third-party individual or business that acts as an intermediary between the tenants and the cloud providers.
Why Cloud Service Brokerage?

1. Helping users determine the best framework for each individual request.

2. A simplified interface, with interoperability benefits – including single sign-on.

3. Cost-effective resources and infrastructure advantages:
   - Including the ability to negotiate technical contracts on-the-fly;
   - Delivering the high levels of flexibility businesses.

4. Enhanced security.
Cloud Service Brokerage

To maximize its own profit, a CSB may distribute tenant requests to clouds which does not efficiently use cloud resources, since maximizing the CSB’s profit does not mean minimizing the cloud providers’ cost.

Q1: Under what pricing policies of cloud providers, when a CSB maximizes its profit, it can also minimize the total energy cost of all cloud providers?

Q2: How should a CSB distribute tenants demands to cloud providers to minimize total energy cost and meanwhile satisfy tenants demands?
Related Work

1. Demand allocation: [Tang, WWW 2007], Sandpiper [Wood, NSDI 2007]
   - Combines CPU and memory consumption into a singular scalar.
   - Using Bin packing algorithms to allocate demands.
   - Drawback: single scalar cannot reflect the consumption of demands and capacity of servers.

   - Auction-style pricing mechanism.
   - Model of cloud bandwidth pricing.
Model

\( M \) demands from tenants:

Cloud service brokerage

\( N \) heterogeneous servers provided by \( L \) cloud providers

\[ v_1 \quad v_2 \quad v_3 \quad \ldots \quad v_M \]

consumption vector

\[ w_i = [w_{i,1}, \ldots, w_{i,K}]^T \]

capacity vector

\[ b_i = [b_{i,1}, \ldots, b_{i,K}]^T \]
Pricing Policy

Indicator variable $x_i$ to represent whether $s_i$ is purchased by the CSB: if yes, $x_i = 1$; otherwise $x_i = 0$.

$z_n = \sum_{s \in C_n} x_i$: the total number of servers that the CSB buys from cloud provider $c_n$.

Indicator variable $y_{i,l}$ to denote whether demand $v_l$ is distributed to cloud server $s_i$: if yes, $y_{i,l} = 1$; otherwise $y_{i,l} = 0$.

The Maximum CSB Profit (MCP) problem:

\[
\begin{align*}
\min \quad & f(\mathbf{z}) = \sum_n L_n(z_n) \\
\text{s.t.} \quad & g_{i,k}(\mathbf{x},\mathbf{y}^i) = \sum_l y_{i,l} w_{i,k} - x_i b_{i,k} \leq 0 \quad \text{for all } i, k \\
& h_l(\mathbf{y}^i) = \sum_i y_{i,l} - 1 = 0, \text{ for all } l \\
& \text{where all } x_i \text{ are integers and all } y_{i,l} \in \{0,1\}.
\end{align*}
\]
Pricing Policy

Denote the energy cost of each $s_i$ by $a_i$.

The min-energy CSB demand allocation (MCD) problem:

\[
\begin{align*}
\text{min} & \quad f(x) = \sum_i a_i x_i \\
\text{s.t.} & \quad g_{i,k}(x,y^i) = \sum_l y_{i,l} w_{i,k} - x_i b_{i,k} \leq 0 \quad \text{for all } i, k \\
& \quad h_i(y) = \sum_l y_{i,l} - 1 = 0, \quad \text{for all } l \\
& \quad \text{where all } x_i \text{ are integers and all } y_{i,l} \in \{0,1\}. 
\end{align*}
\]

Question: under what conditions, the two problems are equivalent?
Pricing Policy

Conclusion: MCP and MCD have the same optimal solution if only if \( L_n(z) = \beta a_n z \), for all \( i \) where \( \beta > 0 \) is a constant.

To encourage a profit-driven CSB to minimize the total energy cost of multiple clouds, the price of each server should be proportional to its energy cost.

If a pricing policy is not proportional to the energy cost, it cannot encourage a profit-driven CSB to minimize the energy cost.
Resource Allocation

- Bin packing
- NP-hard
- Solutions: First Fit Decreasing (FFD) and Best Fit Decreasing (BFD).

In each iteration, BFD picks up the largest unallocated object, and finds out the bin that has the least amount of space left after allocating the object.

- Vector bin packing: extension from 1 dimension to \( n \) dimension
Resource Allocation

Apply FFD or BFD to Vector the bin packing problem (the resource allocation problem).

- Map consumption vector into a single scalar:
  
  BFDSum: \([w_{i,1}, \ldots, w_{i,K}]^T \rightarrow \sum_k w_{i,K}\)
  
  BFDProd: \([w_{i,1}, \ldots, w_{i,K}]^T \rightarrow \prod_k w_{i,K}\)

- Drawback:
  
  1. A single scalar cannot accurately reflect a server’s ability to fit a demand because one type of resource may become the bottleneck.
  2. Omit the energy cost of servers.
Resource Allocation

Balanced Fit Decreasing (BaFD)

**Basic idea**: to iteratively find the demands that make each server’s resource utilization be most balanced, i.e., minimizing the variance of the allocated resources of the selected server, and then picks up the demand that leads to the highest efficiency of the server until its remaining resources cannot hold any existing unallocated demand.

Here, we define the energy efficiency of server $s_i$ by

$$e(s_i) = \frac{\text{the sum of the weights of all demands placed in } s_i}{\text{energy cost of } s_i}$$
Resource Allocation

**Part 1.** temporarily determine the demands that can be allocated to each server to fully utilize its resources.

**Part 2.** choose the server that leads to the highest efficiency value for the actual demand allocation.
Approximation Algorithm

The MCD problem:

\[
\begin{align*}
\text{min} & \quad f(x) = \sum_i a_i x_i \\
\text{s.t.} & \quad g_{i,k}(x,y^i) = \sum_l y_{i,l} w_{i,k} - x_i b_{i,k} \leq 0 \quad \text{for all } i, k \\
& \quad h_l(y_l) = \sum_i y_{i,l} - 1 = 0 \quad \text{for all } l \\
\end{align*}
\]

where all \( x_i, y_{i,l} \in \{0,1\}. \)

MCD-relaxation:

\[
\begin{align*}
\text{min} & \quad f(x) = \sum_i a_i x_i \\
\text{s.t.} & \quad g_{i,k}(x,y^i) = \sum_l y_{i,l} w_{i,k} - x_i b_{i,k} \leq 0 \quad \text{for all } i, k \\
& \quad h_l(y_l) = \sum_i y_{i,l} - 1 = 0 \quad \text{for all } l \\
\end{align*}
\]

where all \( x_i, y_{i,l} \in [0,1]. \)

MCD-relaxation can be directly solved using the simplex method.
Approximation Algorithm

Rounding the fractional solution to integers:

Rounding $y$: for each vector $\mathbf{y}_l = [y_{1,l}, ..., y_{N,l}]$, we set each $y_{i,l}$ by 1 if $y_{i,l} = \{y_1,l, ..., y_{N},l\}$, or 0 elsewhere;

Rounding $x$: first update $x$ value according to the modified $y$ value, and then up round $x$.

Notice: if $s_i$ and $s_j$ have the same capacity vector, and the sum of $x_i$ and $x_j$ is no larger than 1, then we can combine $x_i$ and $x_j$.

The approximation algorithm achieves a constant approximation ratio.
Simulation

**Compared method:**

BFDSum and Sandpiper: all servers’ capacity vectors and demands’ consumption vectors are mapped into singular scales.

**Metrics:**

Total server cost: the total energy consumption of all the servers.

CPU utilization: the percentage of CPU resource used.

Memory utilization: the percentage of memory resource used.
Simulation

Total server cost: Approx and BaFD < BFDSum and Sandpiper
CPU and memory utilization: Approx and BaFD > BFDSum and Sandpiper
Conclusion

1. We found a pricing policy from cloud providers to the CSBs, s.t. maximizing a CSB’s profit is equivalent to minimizing the energy cost of cloud providers.

2. We formulated a demand allocation problem, namely Min-energy CSB Demand allocation problem (MCD).

3. We then devised a greedy algorithm and further proposed an approximation algorithm using LP-relaxation.

4. The experimental results demonstrated the superior performance of our algorithms in both energy efficiency and resource utilizations.

Future work:

1. Consider the scenario where tenants’ demands change over.

2. We will discuss how to apply our technique directly between the cloud providers and the tenants, who do not have enough information of cloud providers (i.e., servers’ capacity vectors) as CSBs.
QUESTIONS ???

Thank you!
Questions & Comments?

Chenxi Qiu, Haiying Shen, Liuhua Chen
Pervasive Communication Laboratory
Clemson University