A Learning Automata based Dynamic Resource Provisioning in Cloud Computing Environments

Authors:
Hamid Reza Qavami
Shahram Jamali
Mohammad Kazem Akbari
Bahman Javadi

Presenting By:
Dr. Bahman Javadi
Western Sydney University, Australia
Resource Provisioning Necessity

• Cloud Resources are:
  – When ever
  – Where ever
  – Resource Pool

• But, let's put on the pink glasses
  – There is not such infinite resource pool!
  – And also when and where ever available
  – Costs, Energy, Environment ...
Dynamic Provisioning
The Efficient Provisioning

- Energy
- Utilization
- Cost

Static Solution
Cloud based solution
Problem Statement

• Minimizing used VMs for application
  – For
    • Cost optimization
    • Utilization
  – While
    • Keeping QoS and SLA parameters

\[
\min\left( \sum_{n=1}^{\text{MaxVM}} VMlist_n^W \right)
\]

\[
\sum_{1}^{\text{MaxOnlineVMs}} MIPS_{VirtualMachines} > \sum_{1}^{\text{CurrentCloudLetNumber}} MI_{CloudLets}
\]
Dynamic Resource Provisioning

- Proposed machine learning approach:
  - Dynamic Resource Provisioning
  Using Learning Automata
Learning Automata

\[ LA \equiv \{\alpha, \beta, p, T\} \]
Defining States

• 3 outputs (states) for L.A
  – Resource increase
    • More resources would be needed
  – Resource decrease
    • Less resources would be needed
  – No changes
    • Current resources would be just enough

\[ \alpha \equiv \{\alpha_1, \alpha_2, \alpha_3\} \]
Feedback

• Average VMs utilization as the feedback
  – It is simply observable from VMM or the VM itself
  – Less monitoring overhead
  – Informative
    • Give us a good status about resources comparing to load

\[
VMs \text{ Avg. Utilization}^w = \frac{\sum_{i=1}^{MaxOnlineVMs} VM_i^w Utilization}{MaxOnlineVMs}
\]
Learning Algorithm
Responding to Last (i\text{th}) Action

• If \((c_i = 0)\) then
  – Reward \(P_i\) & Punish the others

\[
p_i(n+1) = p_i(n) + a \times (1 - p_i(n))
\]
\[
p_j(n+1) = (1 - a) p_j(n) \quad \forall j, \ j \neq i
\]

• If \((c_i = 1)\) then
  – Punish \(P_i\) & Reward the others

\[
p_i(n+1) = (1 - a) p_i(n)
\]
\[
p_j(n+1) = \frac{a}{r-1} + (1 - a) p_j(n) \quad \forall j, \ j \neq i
\]
Novel Intensity Control System

• **Slow convergence** rate in approaches
  – Not enough states for decision
  – More states, make it slower
Experimental Setup

- A scenario in 4 different experiments (using CloudSim):
  - CPU intensive Workload
  - Maximum VM number
    - 20
  - VM processing Core(s)
    - 1 core
  - Core processing power
    - 400 MIPS
  - VM RAM
    - 512 MB
Experimental Results

![Graph showing experimental results](image-url)

- **Dynamic Provisioning Using Proposed System (LAVMP)**
- **Dynamic Provisioning Using (SVMP)**
- **Mean Static Provisioning**
- **Max Static Provisioning (Over Provisioning)**
- **Workload**
Higher Utilization, Lower Cost
## Conclusions

<table>
<thead>
<tr>
<th>Defining Resource Provisioning problem</th>
<th>In Application layer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cost &amp; QoS aware</td>
</tr>
<tr>
<td>Implementing &amp; developing</td>
<td>Dynamic VM provisioning ability in Cloudsim</td>
</tr>
<tr>
<td>Our Solution for defined problem</td>
<td>Dynamic Resource Provisioning</td>
</tr>
<tr>
<td></td>
<td>Using Learning Automata (LAVMP)</td>
</tr>
<tr>
<td>Covering the convergence lack with</td>
<td>Novel aggression control system</td>
</tr>
<tr>
<td>For utilizing VMs</td>
<td>A dispatcher (Job scheduler)</td>
</tr>
<tr>
<td>Achievements</td>
<td>Cost is reduced</td>
</tr>
<tr>
<td></td>
<td>While QoS parameters are considered and improved</td>
</tr>
</tbody>
</table>
Thank You For Your Attention

It doesn’t matter how many resources you have if you don’t know how to use them, they will never be enough.