Reliable and Energy-Efficient Resource Provisioning and Allocation in Cloud Computing

Yogesh Sharma, Bahman Javadi, Weisheng Si
School of Computing, Engineering and Mathematics
Western Sydney University, Australia

Daniel Sun
Data61-CSIRO, Australia
Agenda

1. Introduction
2. Reliability Model
3. Task Execution Model
4. Energy Model
5. Resource Provisioning and Allocation Policies
6. System Architecture
7. Simulation Configuration Parameters
8. Results and Conclusions
• Critical challenge in Cloud Computing environments.

• Service failures have huge impact on service providers such as:
  - Business Disruption
  - Lost Revenues
  - Customer Productivity Loss
Cost of Cloud Outage

* Ref: Calculating the Cost of Data Center Outages, Ponemon Institute© Research Report, 2016
Energy Consumption

Data centers consumption will reach 300 billion kWh in U.S. and 1012.02 billion kWh worldwide by 2020.
Energy Cost and Carbon Footprint

Electricity bill accounts for 20% of a US data center’s Total Cost of Ownership (TCO).

Cloud based data centers in U.S. emit 100 million metric tonne of carbon content each year and will increase to 1034 metric tonne by 2020.
Reliability and Energy-Efficiency Trade-off

The diagram illustrates the trade-off between reliability and energy efficiency. As resource redundancy increases, reliability increases but energy efficiency decreases. Conversely, as VM consolidation increases, energy efficiency increases but reliability decreases. The optimal point is where reliability and energy efficiency are balanced, indicating the trade-off zone where both aspects are considered.
Reliability Model

- System utilization/activity and occurrence of failures are correlated.

- Linear hazard rate/failure rate directly proportional to the utilization following Poisson distribution is

\[ \lambda_{ij} = \lambda_{max_j} u_i^\beta \]

- \( \lambda_{max_j} \): Hazard rate at maximum utilization, \( u_{max_j} \) of a node \( j \)

- \( MTBF_{max_j} \): MTBF at maximum utilization

\[ \lambda_{max_j} = \frac{1}{MTBF_{max_j}} \]
Reliability Model

• Probability (Reliability) with which $vm_i$ running on node $n_j$ with utilization $u_j$ with hazard rate $\lambda_{ij}$ will finish the execution of a task $t_i$ of length $l_i$ is

$$R_{vm_{ij}} = e^{-(\lambda_{ij})l_i}$$

• Probability with which a node $n_j$ will finish the execution of all the $m$ running VMs

$$R_j = \prod_{i=1}^{m} R_{vm_{ij}}$$
Finishing Time with Checkpointing

\[ T' = \sqrt{2 \times T'' \times MTBF_j} \]

- \( T' \): Checkpoint Interval
- \( T'' \): Checkpoint overhead i.e. time taken to save a checkpoint
- \( T^* \): Duration of a lost part of a task that needs to be re-executed
- \( T^# \): Part of the task executed before the occurrence of failure
- \( N'_{ij} \): Number of Checkpoints before a failure on a node \( n_j \) for task \( t_i \)
Finishing Time with Checkpointing

- $N'_{ij}$: Number of Checkpoints before a failure on a node $n_j$ for task $t_i$

\[
N'_{ij} = \left\lfloor \frac{T_{ij}^\#}{T_j^'} \right\rfloor
\]

- Length of the Lost part, $T^*$ will be calculated as

\[
T_{ij}^* = \left( \frac{T_{ij}^\#}{T_j^'} - N'_{ij} \right) * T_j^'
\]

- Finishing Time of a task after the occurrence of $n$ failures under checkpointing scenario will be calculated as the sum of $N'_{ij}, T_{ij}^*$ and time to return (TTR).

\[
T_{ij}^\$ = \begin{cases} 
  l_i + \sum_{k=0}^{n} T_{(ij)k}^* + \left( T'' * \sum_{q=0}^{m} N'_{(ij)q} \right) + \sum_{k=0}^{n} TTR_{(ij)k} , & k, q > 0 \\
  l_i , & Otherwise
\end{cases}
\]
Finishing Time without Checkpointing

- Finishing Time of a task after the occurrence of \( n \) failures under without checkpointing scenario will be calculated as the sum of \( T_{ij}^* \) and time to return (TTR).

\[
T_{ij}^* = \begin{cases} 
  l_i + \sum_{k=0}^{n} T_{(ij)k}^* + \sum_{k=0}^{n} TTR_{(ij)k} & , \quad k > 0 \\
  l_i & , \quad Otherwise
\end{cases}
\]
Energy Model

• The proposed power model is a CPU utilization based model while operating at the maximum frequency.

• \( P_{\text{max}} \), \( P_{\text{min}} \) is the maximum and minimum power consumption by a node \( n_j \), respectively. \( \text{frac}_j \) is the fraction of \( P_{\text{max}} \), \( P_{\text{min}} \).

• The power consumption at utilization \( u_j \) is

\[
P_j(u_i) = (\text{frac}_j \times P_{\text{max}}) + ((1 - \text{frac}_j) \times P_{\text{max}} \times u_i)
\]
Energy Model

- Energy is the amount of power consumed per unit time.

- Energy consumption by a $vm_i$ executing running on a node $n_j$ while executing a task of length $l_i$ in the presence of failures is given as

  $$E_{vm_{ij}} = (P_j(u_i) \times l_i) + E_{waste_{ij}}$$

- $E_{waste_{ij}}$ is the energy wastage because of the failure overheads.
Energy Wastage with Checkpointing

- $E_{\text{checkpoint}}$: Energy consumption while saving checkpoints. Power consumption while saving a checkpoint is $1.15 \times P_{\text{min}}$.

- $E_{\text{re-execute}}$: Energy Consumption while re-executing the lost part of a task because of failures.

\[
E_{\text{waste}_{ij}} = E_{\text{checkpoint}_{ij}} + E_{\text{re-execute}_{ij}}
\]

\[
E_{\text{checkpoint}_{ij}} = \begin{cases} 
1.15 \times P_{\text{min}_{j}} \times \left(T'' \times \sum_{q=0}^{m} N'_{(ij)q}\right), & q > 0 \\
0, & \text{otherwise}
\end{cases}
\]

\[
E_{\text{re-execute}_{ij}} = \begin{cases} 
P_{j}(u_i) \times \sum_{k=0}^{n} T^*_{(ij)k}, & k > 0 \\
0, & \text{otherwise}
\end{cases}
\]
Energy Wastage with Checkpointing

- $E_{checkpoint}$: Energy consumption while saving checkpoints. Power consumption while saving a checkpoint is $1.15 \times P_{min}$.

- $E_{re-execute}$: Energy Consumption while re-executing the lost part of a task because of failures.

$$E_{waste_{ij}} = E_{checkpoint_{ij}} + E_{re-execute_{ij}}$$

$$E_{checkpoint_{ij}} = \begin{cases} 
1.15 \times P_{min} \times \left( T'' \times \sum_{q=0}^{m} N'(ij)_{q} \right), & q > 0 \\
0, & \text{otherwise}
\end{cases}$$

Energy wastage without checkpointing

$$E_{re-execute_{ij}} = \begin{cases} 
P_{j(u)} \times \sum_{k=0}^{n} T^*_{(ij)k}, & k > 0 \\
0, & \text{otherwise}
\end{cases}$$
Resource Provisioning and VM Allocation

Four resource provisioning and VM allocation line algorithms have been proposed.

- Reliability Aware Best Fit Decreasing (RABFD)
- Energy Aware Best Fit Decreasing (EABFD)
- Reliability-Energy Aware Best Fit Decreasing (REABFD)

As a baseline policy Opportunistic Load Balancing (OLB) or Random policy has been used.
Reliability Aware Best Fit Decreasing (RABFD)

- All VMs will be sorted in decreasing order according to their utilization.
- All physical resources will be sorted in increasing order according to their current hazard rate corresponding to the current utilization.
- VM with highest utilization level will get allocated to resource with minimum current hazard rate.

---

Reliability Aware Best Fit Decreasing (RABFD)

Function RELIABILITYAWARE(R)

1. for all j ∈ R do
2. \( \lambda_j \leftarrow r_j . \text{calculateCurrentHazardRate}() \)
3. end for
4. for all j ∈ R do
5. \( R_{\text{sorted}} \leftarrow \lambda_j . \text{sortHazard-rateIncreasing}() \)
6. endfor
7. return \( R_{\text{sorted}} \)
Energy Aware Best Fit Decreasing (EABFD)

- All VMs will be sorted in decreasing order according to their utilization.
- All physical resources will be sorted in increasing order according to their current power consumption corresponding to the current utilization.
- VM with highest utilization level will get allocated to the resource with minimum current power consumption.

---

**Energy Aware Best Fit Decreasing (EABFD)**

Function ENERGYAWARE(R)

1. for all $j \in R$ do
2. $P_j \leftarrow r_j \cdot \text{calculateCurrentPowerConsumption}()$
3. end for
4. for all $j \in R$ do
5. $R_{\text{sorted}} \leftarrow P_j \cdot \text{sortPowerIncreasing}()$
6. endfor
7. return $R_{\text{sorted}}$
Reliability and Energy Aware Best Fit Decreasing (REABFD)

• The ratio of MTBF and power consumption has been used to rank each resource.

• All physical resources will be sorted in decreasing order according to the ratio.

• VM with highest utilization level will get allocated to the resource with highest ratio.

```
Reliability and Energy Aware Best Fit Decreasing (REABFD)

Function RELIABILITYANDENERGYAWARE(R)
1. for all j ∈ R do
2.    MTBF_j ← r_j.calculateCurrentMTBF()
3.    P_j ← r_j.calculateCurrentPowerConsumption()
4.    Ψ_j ← (MTBF_j)/(P_j)
5. end for
6. for all j ∈ R do
7.    R_sorted ← Ψ_j.sortMTBFPowerRatioIncreasing()
8. endfor
9. return R_sorted
```
System Architecture

Cloud Service Users/Brokers

Cloud Management Center (CMC)

- Business
  - Accounting & Billing
  - Pricing & Rating
- Reliability/Energy Aware Resource Provisioning
  - SLA Management
  - Energy Management
  - VM Management
  - Fault Management
- Monitoring
  - Service Monitoring
  - Customer Monitoring

Virtual Layer

- VM

Physical Infrastructure
Workload Parameters

To generate workload, Bag of Task (BoT) applications have been considered.

<table>
<thead>
<tr>
<th>SNo</th>
<th>Parameter</th>
<th>Distribution</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Inter-Arrival Time</td>
<td>Weibull</td>
<td>Scale = 4.25, Shape = 7.86</td>
</tr>
<tr>
<td>2.</td>
<td>Number of Tasks per Bag of Task</td>
<td>Weibull</td>
<td>Scale = 1.76, Shape = 2.11</td>
</tr>
<tr>
<td>3.</td>
<td>Average runtime per Task</td>
<td>Normal</td>
<td>Mean = 2.73, SD = 6.1</td>
</tr>
</tbody>
</table>
Failure Generation Parameters

- Real Failure Traces have been used to add Failures in simulated cloud computing systems.

- Failure information has been gathered from Failure Trace Archive (FTA)

- FTA is a public repository that has failure traces of different architectures gathered from 26 different sites.

- In this work, LANL traces gathered from Los Alamos National Laboratory between 1996-2005 have been used.
Physical Node Parameters

- To gather power profiles of the physical machines, spec2008 benchmark has been used.
- Node type has been chosen on the basis of the node information provided in the failure traces.

<table>
<thead>
<tr>
<th>SNo</th>
<th>Node Type</th>
<th>Cores</th>
<th>Memory (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Intel Platform SE7520AF2 Server Board</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>2.</td>
<td>HP ProLiant DL380 G5</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>3.</td>
<td>HP ProLiant DL758 G5</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>4.</td>
<td>HP ProLiant DL560 Gen9</td>
<td>128</td>
<td>128</td>
</tr>
<tr>
<td>5.</td>
<td>Dell PowerEdge R830</td>
<td>256</td>
<td>256</td>
</tr>
</tbody>
</table>
Average Reliability

- The reliability with which application has been executed on provisioned resources.

**REABFD vs other policies**

<table>
<thead>
<tr>
<th>Policy</th>
<th>Checkpointing</th>
<th>Without Checkpointing</th>
</tr>
</thead>
<tbody>
<tr>
<td>RABFD</td>
<td>5%</td>
<td>6%</td>
</tr>
<tr>
<td>OLB</td>
<td>16%</td>
<td>15%</td>
</tr>
<tr>
<td>EABFD</td>
<td>17%</td>
<td>23%</td>
</tr>
</tbody>
</table>

- Policies using checkpointing gives better reliability by 5% to 9% than without checkpointing.
Average Energy Consumption

- Energy consumption incurred by the provisioned resources.

### REABFD vs other policies

<table>
<thead>
<tr>
<th>Policy</th>
<th>Checkpointing</th>
<th>Without Checkpointing</th>
</tr>
</thead>
<tbody>
<tr>
<td>RABFD</td>
<td>7%</td>
<td>7%</td>
</tr>
<tr>
<td>OLB</td>
<td>50%</td>
<td>15%</td>
</tr>
<tr>
<td>EABFD</td>
<td>61%</td>
<td>50%</td>
</tr>
</tbody>
</table>

- Policies using checkpointing consumes more energy by 2% to 5% than without checkpointing.
Average Energy Consumption

- Energy consumption incurred by the provisioned resources.

**REABFD vs other policies**

<table>
<thead>
<tr>
<th>Policy</th>
<th>Checkpointing</th>
<th>Without Checkpointing</th>
</tr>
</thead>
<tbody>
<tr>
<td>RABFD</td>
<td>7%</td>
<td>7%</td>
</tr>
<tr>
<td>OLB</td>
<td>50%</td>
<td>15%</td>
</tr>
<tr>
<td>EABFD</td>
<td>61%</td>
<td>50%</td>
</tr>
</tbody>
</table>

In-fact, better not to use any policy and keeps allocation random, if reliability will not be considered.
Average Energy Wastage

- The amount of energy wasted because of the failure overheads.

### REABFD vs other policies

<table>
<thead>
<tr>
<th>Policy</th>
<th>Checkpointing</th>
<th>Without Checkpointing</th>
</tr>
</thead>
<tbody>
<tr>
<td>RABFD</td>
<td>8%</td>
<td>11%</td>
</tr>
<tr>
<td>OLB</td>
<td>53%</td>
<td>54%</td>
</tr>
<tr>
<td>EABFD</td>
<td>67%</td>
<td>70%</td>
</tr>
</tbody>
</table>

- Wastage has been observed more by 36% in the absence of checkpointing because of large re-execution overheads.
Average Turnaround Time

- It is the time taken by each task of BoT application to finish.

**REABFD vs other policies**

<table>
<thead>
<tr>
<th>Policy</th>
<th>Checkpointing</th>
<th>Without Checkpointing</th>
</tr>
</thead>
<tbody>
<tr>
<td>RABFD</td>
<td>7%</td>
<td>7%</td>
</tr>
<tr>
<td>OLB</td>
<td>39%</td>
<td>39%</td>
</tr>
<tr>
<td>EABFD</td>
<td>46%</td>
<td>46%</td>
</tr>
</tbody>
</table>

- Better turnaround time has been achieved by 7% while using checkpointing.
Deadline-Turnaround Time Fraction

- It is the margin by which the turnaround time has been exceeded from the deadline.

### REABFD vs other policies

<table>
<thead>
<tr>
<th>Policy</th>
<th>Checkpointing</th>
<th>Without Checkpointing</th>
</tr>
</thead>
<tbody>
<tr>
<td>RABFD</td>
<td>3%</td>
<td>6%</td>
</tr>
<tr>
<td>OLB</td>
<td>6%</td>
<td>7%</td>
</tr>
<tr>
<td>EABFD</td>
<td>15%</td>
<td>20%</td>
</tr>
</tbody>
</table>

- For scenarios without checkpointing, the makespan has been exceeded more by 7% in comparison to checkpointing.

- Re-execution has been found higher by 36% for without checkpointing scenario.
Average Benefit Function

- It is ratio of reliability and energy consumption of the system.

**REABFD vs other policies**

<table>
<thead>
<tr>
<th>Policy</th>
<th>Checkpointing</th>
<th>Without Checkpointing</th>
</tr>
</thead>
<tbody>
<tr>
<td>RABFD</td>
<td>29%</td>
<td>34%</td>
</tr>
<tr>
<td>OLB</td>
<td>76%</td>
<td>85%</td>
</tr>
<tr>
<td>EABFD</td>
<td>82%</td>
<td>78%</td>
</tr>
</tbody>
</table>

- Scenarios using checkpointing gives better benefit function upto 14% than without checkpointing.
Conclusion and Future Work

• While giving emphasis only to the energy optimization without considering reliability factor, results are contrary to the expectation.

• More energy consumption has been experienced due to the energy losses incurred because of failure overheads.

• Reliability-Energy Aware Best Fit Decreasing (REABFD) policy outperforms all the other policies.

• It has been revealed that by considering both energy and reliability factors together, both factors can be improved better than being regulated individually.

• In future, machine learning methods will be used to predict the occurrence of failures.

• By using failure prediction results, VM migration and consolidation mechanism will be adopted to further optimized the fault tolerance and energy consumption.
Thank You