Load Balancing

- Load Balancing Mechanisms in Data Center Networks
- Load balancing vs. distributed rate limiting: an unifying framework for cloud control
- Load Balancing for Internet Distributed Services using Limited Redirection Rates
LOAD BALANCING MECHANISMS IN DATA CENTER NETWORKS

Authors: Santosh Mahapatra, Xin Yuan

Publisher: Department of Computer Science, Florida State University

Present: Dao-Yuan Chang
OUTLINE

1. Introduction
2. PROPOSED LOAD BALANCING SCHEMES
3. Performance Analysis
4. Conclusion
INTRODUCTION

- VLB technique can be applied at packet-level and flow level.
- To avoid the out-of-order packet issue in a data center network with TCP/IP communications, VLB can only be applied at the flow level.
  - all packets in one flow use the same path
INTRODUCTION

- While flow-level VLB performs well on uniform traffics, it does not deliver high performance for non-uniform traffics level.
- As the message size increases, packet level VLB becomes more effective in comparison to flow level VLB.
  - Flow-level VLB has performance issues when it routes multiple flows into one link and causes congestion.
Fig. 5. Impact of message size ($d_i = d_o = 12$, $u_{tor} = 2$, $npr = 8$, $bw_{sw} = 40Gbps$, $bw_{ms} = 10Gbps$)
1. Introduction
2. PROPOSED LOAD BALANCING SCHEMES
3. Performance Analysis
4. Conclusion
PROPOSED LOAD BALANCING SCHEMES

- Two alternate load balancing schemes to overcome the problems in the flow-level VLB:
  - 1) Queue-length directed adaptive routing
  - 2) Probing based adaptive routing
It works by selecting the output port with the smallest queue length that can reach the destination.

- The first packet of a flow adaptively constructs the path towards the destination based on the queue-lengths of output ports.
PROBING BASED ADAPTIVE ROUTING

- It probes the network by sending a number of probe packets following different paths to the destination.
- The receiving node replies with an acknowledgment packet for the first probe packet received and drops the other probe packets.
  - *The acknowledgment packet carries the path information, and the source will then use the selected path for the communication.*
OUTLINE

1. Introduction
2. PROPOSED LOAD BALANCING SCHEMES
3. Performance Analysis
4. Conclusion
Fig. 9. Performance of the proposed load balancing schemes for clustered traffics on balanced systems with different bandwidth ratios ($d_i = d_a = 16, u_{tor} = 2, bw_{ms} = 1Gbps, msize=1000000$)
**Performance Analysis**

Fig. 11. Improvement percentage of probing based adaptive routing over flow-level VLB for clustered traffics on balanced systems with different bandwidth ratios ($d_i = d_a = 16$, $u_{tor} = 2$, $bw_{ms} = 1Gbps$, msize=1000000)
Fig. 12. Packet delays for clustered traffics on balanced systems with different bandwidth ratios ($d_i = d_a = 16$, $u_{tor} = 2$, $bw_{ms} = 1Gbps$, $m_{size}=1000000$)
OUTLINE

1. Introduction
2. PROPOSED LOAD BALANCING SCHEMES
3. Performance Analysis
4. Conclusion
**CONCLUSION**

- Proposed two schemes achieve better performance in the cases when flow-level VLB is not effective.

- Including our proposed schemes, the results indicate that flow-level load balancing schemes significantly worse than packet-level VLB.
If the shortest path destination sever is in heavy loading state, the performance will be degraded.
LOAD BALANCING VS. DISTRIBUTED RATE LIMITING:
AN UNIFYING FRAMEWORK FOR CLOUD CONTROL

Rade Stanojević, Robert Shorten
Hamilton Institute, NUIM, Ireland

Student：呉星佑
GOAL

- Control cloud based services for load balancing and distributed rate limiting

- Find the matrix of demand allocations $x$ and vector of server capacities for which the performance at each server is (approximately) equal and that minimizes the aggregate cost

$$V(x) = \sum_{j=1}^{M} \sum_{i=1}^{N} C_{ij} x_{j}^{(i)} + \sum_{i=1}^{N} D_{i}(y_{i}).$$
**Basic Assumption**

- Cloud control in a model does not assume infinite capacity
- A job can be split in arbitrary manner
- Communication between servers is performed via small UDP packets
**PSEUDO CODE OF CARTON**

1. **DRL-UpdateCapacities()**
   2. Once every $\Delta$ units of time do
   3. for $i = 1 : N$
   4. \[ \mu_i \leftarrow \mu_i + \eta \sum_{(i,j) \in E} (q_i - q_j) \]
   5. endfor
   6. enddo

7. **InitializeCapacities()**
   8. for $i = 1 : N$
   9. \[ \mu_i \leftarrow \frac{\mu}{N} \]
   10. endfor

11. **UpdateDemands()**
   12. $g(k) = \text{Subgradient of } V \text{ in } x(k)$
   13. \[ x(k + 1) = x(k) - \frac{1}{\sqrt{k}}g(k) \]
First, subgradient method that allocates the workload $x$ that the cost function is minimized.

Second, DRL algorithm that allocates the server capacities that ensures performance levels at all the servers are equal.
SUBGRADIENT METHOD (1/2)

- A iterative methods for solving convex minimization problems
- Applied to a non-differentiable objective function
- Choice step-size $\alpha$ for convergence
Subgradient Method (2/2)

\[ x(k+1) = \hat{x}(k) - \frac{1}{\sqrt{k}} g(k) \]

- \( \alpha \) is \( 1/\sqrt{k} \)
- Finally, because \( \alpha \) is convergence, the workload \( x \) is constant
- So the \( x \) is the optimal
Theorem 1: Let $l_i$ be the degree of limiter $i$ in the communication graph $G$. Then if $\eta$ satisfies:

$$0 < \eta < \min_{1 \leq i \leq N} \frac{1}{l_i},$$

then for all $i, j$

$$\lim_{k \to \infty} q_i(k) - q_j(k) = 0.$$
By this Theorem, $\eta$ ensures the performance indicator $q$ between $i$ and $j$ will be constant eventually.

- It means the capacity $\mu$ is equal.
DISTRIBUTED RATE LIMITING

- q is a performance indicator, include available bandwidth, mean response time, etc.

- If the q at limiter i is higher than q at j, the some extra capacity should be allocated to limiter i which should be compensated by reducing the capacity of limiter j
Case 1. \( D_i \equiv 0 \)
Case 1. $D_i \equiv 0$

Fig. 3. $D_i \equiv 0$: steady-state. Each line represents a job-server pair with thickness proportional to $(x_j^{(3)})^*$. The diameter of each circle is $\mu_i^*$. 
Case 2. \( D_i(y) = y(y - \frac{\mu}{N})^+ \)
Case 2. $D_i(y) = y(y - \frac{\mu_i}{N})^+$

Fig. 5. $D_i(y) = y(y - \frac{\mu_i}{N})^+$: steady-state. Each line represents a job-server pair with thickness proportional to $(x_j^{(i)})^*$. The diameter of each circle is $\mu_i^*$. 
DYNAMIC DEMANDS
Dynamic Demands
Load balancing has been recently challenged by the DRL to minimize the cost.

This mechanism is simple, easy to implement and has very low computation and communication overhead.
In fact, I don’t know whether this is a good model because it is a static environment

In real life, request is in and out frequently

More important is how long to convergence and must close to load balancing
### SYMBOL TABLE

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>number of servers</td>
</tr>
<tr>
<td>$M$</td>
<td>number of jobs</td>
</tr>
<tr>
<td>$\mu$</td>
<td>the aggregate capacity</td>
</tr>
<tr>
<td>$\mu_i$</td>
<td>capacity at server $i$</td>
</tr>
<tr>
<td>$q_i$</td>
<td>performance indicator at server $i$</td>
</tr>
<tr>
<td>$d_j$</td>
<td>demand of job $j$</td>
</tr>
<tr>
<td>$x_j^{(i)}$</td>
<td>fraction of job $j$ served by server $i$</td>
</tr>
<tr>
<td>$y_i$</td>
<td>traffic intensity at server $i$</td>
</tr>
<tr>
<td>$C_{i,j}$</td>
<td>cost of serving a unit of job $j$ at server $i$</td>
</tr>
<tr>
<td>$D_i(y)$</td>
<td>cost of using server $i$ with demand $y$</td>
</tr>
<tr>
<td>$JFI$</td>
<td>Jain’s fairness index</td>
</tr>
<tr>
<td>$\eta$</td>
<td>gain parameter</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>load</td>
</tr>
</tbody>
</table>

**TABLE I**  
SYMBOL MAP
LOAD BALANCING FOR INTERNET DISTRIBUTED SERVICES USING LIMITED REDIRECTION RATES

Source: Dependable Computing (LADC), 2011
5th Latin American Symposium on
Author: Alan Massaru Nakai, Edmundo Madeira, and Luiz E. Buzato
Present: Wei-Cheng Chen
OUTLINE

* Introduction
* Related Work
* Limited Redirection Rates
  * Virtual Web Resource
  * Load Balancing Heuristics
* Evaluation
* Conclusion
The web has become the universal support for applications. Increasingly, heavy loaded applications.

Good performance in these environments is the efficiency of the load balancing mechanism used to distribute client requests among the replicated services.
There are four classes of load balancing mechanisms for geographically distributed web services:

- DNS-based
- Server-based
- Dispatcher-based
- Client-based
INTRODUCTION (3/3)

* **DNS-based**
  * the Authoritative DNS (ADNS) of the replicated web server performs the role of the client request scheduler.

* **Server-based**
  * the load balancing policy runs in the server side. *overloaded server replica can redirect requests to other replicas.*

* **Dispatcher-based**
  * a host placed between clients and server replicas receives all requests and forwards them to the appropriate replica.

* **Client-based**
  * The client runs the distribution load policy and decides to which server it sends the requests.
**Related Work (1/2)**

* Focus on server-based load balancing solutions.
  * Performance of the load balancing mechanisms
    * Centralized vs. Distributed control

* Conclusion
  * a trade-off between the
    * slightly better performance of centralized algorithms
    * the lower complexity of distributed ones.

(performance = response time)
Problem:

* none of the existing proposals focus on techniques that prevent redirected requests to overload the remote server.

Our solution avoids this problem by

1. A new strategy based on limited rates of request redirection
2. A heuristic based on this mechanism that tolerates abrupt load peaks.

Also tries to minimize the system response time considering the load status of the web servers and network latencies.
**LIMITED REDIRECTION RATES (1/8)**

- The best choice would be to redirect requests to the closest lightly loaded web server.
  - This intuitive policy may be very inefficient if the redirections overload the remote web server.

**Example:**

Server capacity: 100 rps.

- A: redirect to B
- B: overloaded
- A: redirect to C
- C: overloaded

*Figure 1. Intuitive policy.*
The main contributions of our work are:

- A protocol for limited redirection rates that avoids the overloading of the remote servers.
- A middleware that supports this protocol and minimizes the response times of redirected requests.
- A heuristic based on the protocol that tolerates abrupt load changes.
**LIMITED REDIRECTION RATES (3/8)**

- Limited redirection rates that avoids the overloading of the remote servers.
  - A protocol that imposes a maximum limit of requests that each overloaded server is allowed to redirect to each lightly loaded web server.

- We call this protocol LRR (Limited Redirection Rate)

![Figure 3. Limited Redirection Rate.](image-url)
**LIMITED REDIRECTION RATES (4/8)**

* A middleware that supports this protocol and minimizes the response times of redirected requests.

* Virtual Web Resource (VWR), middleware that allows web servers to share their resources.

* Web servers can assume two different states:
  - **Provider**: Lightly loaded web servers that can share resources with others.
  - **Consumer**: Overloaded web servers that consume resources shared by providers.

* The VWR abstracts the location of the providers to the consumers, which see the resources shared by the providers as a single pool of remote resources.

---

*Figure 4. Virtual web Resource. remote resources.*
The middleware that implements VWR is composed of two software elements.

- VWR-Master and VWR-Nodes.

- **VWRMaster**(core of the middleware)
  - responsible for allocating virtual resources to the consumers and also for reallocating virtual resources when the status of providers changes.

- **VWRNodes (software agents that run on every web server)**
  - They monitor the current load of the web server and communicate with the VRWMaster to offer local resources or to allocate virtual resources.
**LIMITED REDIRECTION RATES (6/8)**

* VWRNode interacts with the VWRMaster through the following messages:
  - **ReportLatencyTable**: sent by VWRNodes to report the estimate of latencies among the local web server and the others.
  - **AcquireVirtualResources**: sent by VWRNodes to request the allocation of an amount of virtual resources
  - **ProviderList**: sent by VWRMaster to inform the list of providers that will provide the virtual resources to the VWRNode.
  - **ReportResourceAvailability**: sent by VWRNodes to announce the amount of resources they are able to share or to update the value amount already shared.
LIMITED REDIRECTION RATES (7/8)

Figure 5. Interaction between a consumer and the middleware.

Figure 6. Interaction between a provider and the middleware.
**LIMITED REDIRECTION RATES (8/8)**

**Load Balancing Heuristics**

- The VWR middleware combined with some policy that decides when to redirect requests.

- The basic idea of this heuristic is trying to keep the local request queue shorter than a threshold.

If a request needs to wait in a queue, it should not pay the price of a redirection latency.

```plaintext
IF Queue <= th_queue, THEN
    Deliver request
    RETURN
ELSE
    Try to send the request to VWR
    IF request was sent to VWR, RETURN
ENDIF

Deliver request
RETURN
```

**Figure 7. Load Balancing Heuristic.**
EVALUATION (1/4)

- **Environment**: six replicas of the web server that are worldwide distributed (Figure 9):
  - run the simulations on 21 variations of the scenario depicted by Figure 9
  - *(1 overload 5 lightly; 2 overload 4 lightly)*

Table 1. Table of latencies among the replicas (in milliseconds).

<table>
<thead>
<tr>
<th></th>
<th>S1</th>
<th>N1</th>
<th>E1</th>
<th>E2</th>
<th>A1</th>
</tr>
</thead>
<tbody>
<tr>
<td>N1</td>
<td>89</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E1</td>
<td>138</td>
<td>48</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E2</td>
<td>140</td>
<td>58</td>
<td>18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A1</td>
<td>193</td>
<td>109</td>
<td>151</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td>272</td>
<td>156</td>
<td>114</td>
<td>122</td>
<td>68</td>
</tr>
</tbody>
</table>

Figure 9. Web Server Replicas.
EVALUATION (2/4)

- compared with two other load balancing policies
  - Round Robin with Asynchronous Alarm (RR)
  - Smallest Latency (SL)

Figure 10. Mean response times for all scenarios.
Analyse in scenario 8

- A frequency analysis shows that while 95% of the requests were responded in less than 0.14 s using LRR (Limited Redirection Rate), using RR and SL, the same percentage of requests were responded in 0.28 and 0.27 s.
In scenario 8, the closest lightly loaded servers for S1 and E1 are distinct servers (N1 and E2 respectively), thus, this scenario prevents SL from choose the same remote server for S1 and E1. As Table 3 shows.

**Table 3. Scenario 8 - Percentage of redirections.**

<table>
<thead>
<tr>
<th>Policy</th>
<th>S1</th>
<th>N1</th>
<th>E1</th>
<th>E2</th>
<th>A1</th>
<th>A2</th>
<th>Total Redir.</th>
</tr>
</thead>
<tbody>
<tr>
<td>RR</td>
<td>0%</td>
<td>24.41%</td>
<td>2.35%</td>
<td>24.41%</td>
<td>24.41%</td>
<td>24.41%</td>
<td>214587</td>
</tr>
<tr>
<td>SL</td>
<td>0%</td>
<td>54.88%</td>
<td>0%</td>
<td>45.12%</td>
<td>0%</td>
<td>0%</td>
<td>214689</td>
</tr>
<tr>
<td>LRR</td>
<td>0%</td>
<td>42.64%</td>
<td>2.34%</td>
<td>21.88%</td>
<td>0.67%</td>
<td>32.46%</td>
<td>434996</td>
</tr>
</tbody>
</table>
CONCLUSION

- We have presented a new server-based load balancing policy for worldwide distributed web servers.
- Our solution was designed on the basis of a protocol that limits the redirection rates, preventing such a problem and reducing response times compared with two other policies.
- Is it worth it?
Question:
What are four classes of load balancing mechanisms for geographically distributed web services?

Ans:
- DNS-based
- Server-based
- Dispatcher-based
- Client-based