International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:11, No:1, 2017

Research on Load Balancing Technology for Web Service Mobile Host

Yao Lu, Xiuguo Zhang, Zhiying Cao

***Abstract*—**In this paper, Load Balancing idea is used in the Web service mobile host. The main idea of Load Balancing is to establish a one-to-many mapping mechanism: An entrance-mapping request to plurality of processing node in order to realize the dividing and assignment processing. Because the mobile host is a resource constrained environment, there are some Web services which cannot be completed on the mobile host. When the mobile host resource is not enough to complete the request, Load Balancing scheduler will divide the request into a plurality of sub-requests and transfer them to different auxiliary mobile hosts. Auxiliary mobile host executes sub-requests, and then, the results will be returned to the mobile host. Service request integrator receives results of sub-requests from the auxiliary mobile host, and integrates the sub-requests. In the end, the complete request is returned to the client. Experimental results show that this technology adopted in this paper can complete requests and have a higher efficiency.

***Keywords*—**Dinic, load balancing, mobile host, web service.

I. Introduction

WEB service technology can run on different machines with different applications without the additional and specialized of third party software or hardware to exchange data or integrate with each other. According to the standard of Web service, data are executed with each other on different languages, platforms, or internal protocols. Web service is a self-described, self-contained, and available network module [1].

Web service technology enables interoperability of distributed systems and has become more and more popular. Now, the mobile host is an indispensable part of people's daily life today. The wireless networks development and the widespread using of Web service provided Web service on the mobile devices are widely popular [2]. Web service of mobile host refers to deploying the service on mobile host or providing mobile host using Web service.

Load Balancing in cluster is a large scale parallel Load Balancing strategy [3]. Load Balancing can be divided into static Load Balancing and dynamic Load Balancing [4]. The common static Load Balancing methods are round-robin method, ratio method, and priority method. Round-robin method is the most commonly static Load Balancing algorithm. The common dynamic Load Balancing methods are the least connections method, the fastest connection method, observation method, prediction method, and the dynamic performance allocation method [5].

Yao Lu, Xiuguo Zhang, Zhiying Cao are with the School of Information Science and Technology, Dalian Maritime University, Dalian, China (e-mail: 609681867@qq.com).

The research on Load Balancing technology for Web service mobile host is not a lot. The task of Remote Web service in the mobile host is to achieve a lightweight framework for Web services host [6]. Compared to the traditional Web service mobile framework, it achieves higher throughput and less resource consumption. The task is divided into the intermediate node, forwarding node, and backed node when the mobile host resources are not enough [7], [8]. But, most of these methods are used on the desktop PC, and these are not applicable to the mobile host.

This paper still uses the point of the Load Balancing algorithm. But, because of the mobile host is a resource constrained environment, the Load Balancing algorithm such as round-method, random-method, and minimum queue-method [9] cannot be fully applied to mobile host. Therefore, based on the characteristics of the mobile host, this paper researches on Load Balancing technology of Web service mobile host.

II. Web Services Mobile Host Load Balancing Framework

A. Main Components of the Framework

The framework is composed of mobile client, mobile host, and multiple auxiliary mobile hosts.

1. Client

The client is the user of the service, which is used to send the request to the mobile host. The auxiliary mobile host is to execute the client's request and to return the result to the client.

This paper makes the studies that each request may have one or more services comprised [10]. The request may contain a single-request or composite-requests. All types of service requests and resources consumed are defined in advance.

If there is only one service in the request, it is called a single-request. Each single-request is defined to an atomic and non-repartition request. When there are more than one service in the request, it is defined as a composite-request.

When a request is a composite-request, Service request analyzer assumes it as having N services, recorded as $S_{1.....}S_N$. S_s records as the service of start, S_2 is the precursor-service of S_3 , S_3 is subsequence-Service of S_2 . S_e records as the end of service. The execution of subsequence-Service is after the precursor-service, which calls the two services to be dependent on each other.

2. Mobile Host

a. Service Request Response Listener

Service request response listener can be divided into service request listener and service response listener. Service request listener receives request and sends request to Service request analyzer. Service response listener receives the execution results from auxiliary mobile host and returns it back.

b. Service Request Analyzer

Service request analyzer receives the request sent by the Service request response listener. First, it stores the request to the service request pool, and the request is stored in the service request form as a queue. The form extracts address, request content, and other information. At least, Service request analyzer sends the information to the Load Balancing scheduler.

c. Load Balancing Scheduler

The main function of the Load Balancing scheduler is to implement Load Balancing algorithm. Load Balancing scheduler determines whether there is the auxiliary mobile host which can complete the request. If there is the auxiliary mobile host which can execute the request, then it sends the request to the auxiliary mobile host. Otherwise, the request is divided into several sub-requests, and the sub-requests are sent to the auxiliary mobile host.

d. Request Transponder

Request transponder receives the request from Load Balancing scheduler, and distributes request to the Load transfer receiver for the auxiliary mobile host.

e. Service Request Back Receiver

Service request back receiver receives the service request result from the auxiliary mobile host, and sends the result to the service request integrator.

f. Service Request Integrator

When the request is the composite-requests, the Service request integrator needs to send the combination of request to the client [11]. Service request integrator receives the sub-requests and returns the results of enforcement to Service request response listener.

g. Resource Monitor

Resource monitor receives the auxiliary mobile host of the resource consumption, and each of the auxiliary mobile host sends the amount of resources consumed to the resource record table.

h. Resource Record Form

Resource record form notes the load capacity, battery, CPU, and other performance of the auxiliary mobile hosts [12], and sends this information to the Load Balancing scheduler.

3. Auxiliary Mobile Host

Auxiliary mobile host cluster contains n auxiliary mobile hosts and is denoted as $A_1 \ldots\ldots A_n\ (n = 4)$ [13].

a. Load Transfer Receiver

Load transfer receiver receives the request sent by the mobile host and sends the service request to the auxiliary mobile host actuator.

b. Auxiliary Mobile Host Actuator

Auxiliary mobile host actuator performs the request sent by the Load transfer receiver, and when executed successfully, it returns the executed result and sends the message to the service return receiver.

B. Web Services Mobile Host Load Balancing Framework

In this paper, the framework of Load Balancing technology for Web Service mobile host shown in Fig. 1.

III. Load Balancing Algorithm

A. Parameter Acquisition

Firstly, Resource monitor gets the auxiliary mobile host resource consumption of P_{RAM} , P_{ROM} , $P_{battery}$, P_{CPU} , and those represent the use of auxiliary mobile host RAM, the use of auxiliary host ROM, auxiliary mobile host power usage, and auxiliary mobile host CPU usage as shown in Table I.

TABLE I
Auxiliary Mobile Host Resource Usage Parameters

Name	Formula
P_{RAM}	$P_{RAM} = \frac{E_{RAM}}{R_{RAM}}$
P_{ROM}	$P_{ROM} = \frac{E_{ROM}}{R_{ROM}}$
$P_{battery}$	$P_{battery} = \frac{E_{battery}}{R_{battery}}$
P_{CPU}	$P_{CPU} = \frac{E_{CPU}}{R_{CPU}}$

According to Table I, it calculates the auxiliary mobile host resource usage $LoadRate$ for the formula $LoadRate = P_{RAM} + P_{ROM} + P_{battery} + P_{CPU}$

B. Algorithm Description

The request sent by client may be a single-request or composite-requests. Service request analyzer determines the type of the request and then sends the request to Load Balancing scheduler.

The Service request analyzer judges if auxiliary mobile host which can complete the service request exists or not. If not, the request will be divided into several sub-requests.

Load Balancing observes the three principles:

1) The sub-requests number should not be too much. Because when dividing the request to more sub-requests, the conversion process will also increase. It then increases the cost of consumption and communication cost between sub-requests.
2) Sub-requests parallelism should be as much as possible. The meaning of parallelism implies that multiple code can be executed at the same time [14]. After the composite-requests are divided into different auxiliary mobile hosts, it

is better to perform the sub-requests at the same time. Parallelly executed sub-requests can reduce the total time of request.

3) Try to reduce the communication between the auxiliary mobile hosts so that it can decrease the communication cost [15].

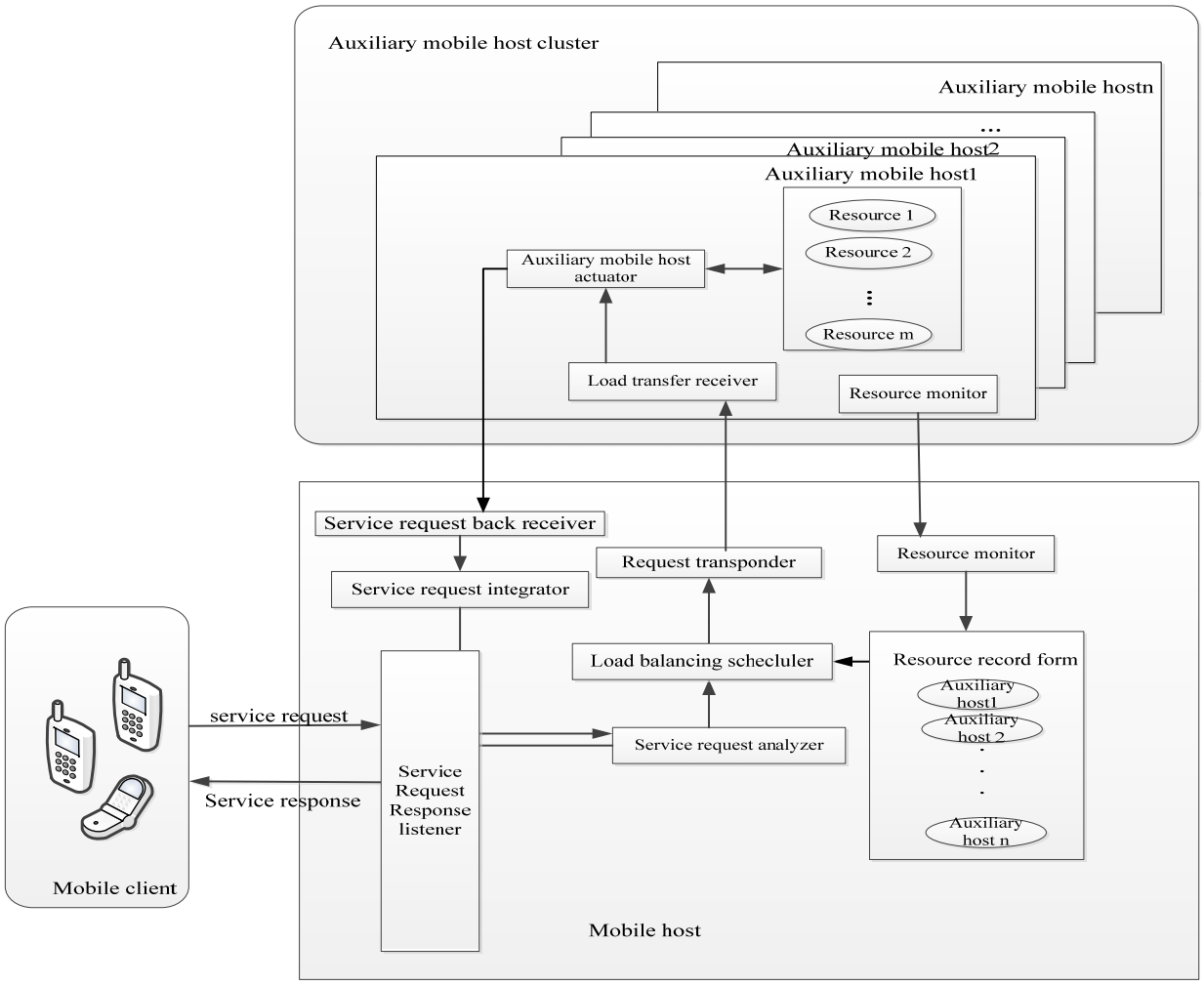

Fig. 1 Load balancing framework of web service mobile host

According to the framework, this section provides a Load Balancing algorithm for Web service mobile host, which is divided into ten steps:

1) Calculate the resources utilization rate of each auxiliary mobile host $L_1 \ldots\ldots L_n$ (n=4).
2) Calculate the amount of resources required from a request sent by the client, recorded as S_{need} . The residual-resource of the Auxiliary mobile host is recorded as $S_{A1} \ldots\ldots S_{An}$.
3) Service request analyzer determines whether there is an Auxiliary mobile host which can perform the request [16]. If there is only one auxiliary mobile host which can execute the request, Request transponder will send the request to the auxiliary mobile host. If there are many auxiliary mobile hosts which can execute the request, those auxiliary mobile hosts search for the residual-resource of auxiliary mobile host which is closest to the S_{need} of the request, and then send the request to the auxiliary mobile host.
4) If the auxiliary mobile host which can execute the request does not exist, Service request analyzer judges whether the request is a single-request or a composite-request. Because single-request's nature is an atomic and non-repartition service, the request that mobile host returns to client cannot be performed. For a composite-request, according to the Dinic minimum cut algorithm [17], the request is divided into two sub-requests, recorded as sub_1 and sub_2 .
5) S_{sub1} is defined as the resources needed of the sub_1 . Assuming that there are the Auxiliary mobile hosts $A_1 \ldots\ldots A_n$, A_x is defined as any auxiliary mobile host. S_{Ax} is defined as the residual-resource of any auxiliary mobile hosts. Looking for the auxiliary mobile host of which $S_{Ax} >= S_{sub1}$, if there is only one, Request transponder

sends sub_1 to the auxiliary mobile host. If there are more than one auxiliary mobile host which can perform the sub_1, Load Balancing scheduler finds the S_{Ax} closest to S_{sub1} and sends to the auxiliary mobile host.

6) If there is no $S_{Ax} >= S_{sub1}$, Service request analyzer traverses the auxiliary mobile host to look for $S_{Ax1} + S_{Ax2} >= S_{sub1}$. If there is just one group of auxiliary mobile hosts such as $Ax_1 + Ax_2$, which can execute the sub_1, Request transponder sends the sub_1 to the auxiliary mobile hosts group. Traversing sub_1 and divide the sub_1 into sub_3 and sub_4, the partition point is the S_{need} of sub_3 which is closest to the auxiliary mobile host of maximum resource. Request transponder sends sub_3 and sub_4 to the auxiliary mobile hosts. If there are a lot of auxiliary mobile hosts group which can execute sub_1, select the auxiliary mobile hosts group such that $S_{Ax1} + S_{Ax2}$ closest to the S_{sub1}, and use the same technology to divide sub_1 to sub_3 and sub_4, and send to the auxiliary mobile host.
7) If there is no $S_{Ax1} + S_{Ax2} >= S_{sub1}$, Service request analyzer traverses the auxiliary mobile host to looking for $S_{Ax1} + S_{Ax2} + S_{Ax3} >= S_{sub1}$. If there is just one group of auxiliary mobile hosts such as $Ax_1 + Ax_2 + Ax_3$ can execute the sub_1, Request transponder will send sub_1 to the auxiliary mobile hosts group. Traverse sub_1 and divide the sub_1 into sub_3, sub_4 and sub_5, the partition point of sub_3 is of which $S_{needsub3}$ is most close to the residual-resource of S_{AX1}.and the partition point of sub_4 is of which $S_{needsub4}$ is most close to the residual-resource of S_{AX2}, and the sub_5 give to the last auxiliary mobile host. Request transponder sends sub_3, sub_4 and sub_5 to the auxiliary mobile hosts. If there are a lot of auxiliary mobile hosts group which can execute sub_1, select the auxiliary mobile host such that $Ax_1 + Ax_2 + Ax_3$ closest to the S_{sub1}, and use the same technology to divide sub_1, and send it to the auxiliary mobile hosts.
8) If there is no $S_{Ax1} + S_{Ax2} + S_{Ax3} >= S_{sub1}$, Load Balancing scheduler calculates $S_{Ax1} + S_{Ax2} + S_{Ax3} + S_{Ax4}$. If $S_{Ax1} + S_{Ax2} + S_{Ax3} + S_{Ax4} >= S_{sub1}$, according to the residual-resource of auxiliary mobile hosts, divide the sub_1 into four parts such that the four parts of the resource are close to the residual-resource of auxiliary mobile host. The four parts are recorded as sub_3, sub_4, sub_5 and sub_6 respectively which are then sent to the corresponding auxiliary mobile hosts.
9) If the sum of the residual-resource of all auxiliary mobile hosts cannot complete the request, according to the Dinic algorithm, the sub_1 is divided the into two sub-requests again in accordance with the algorithm to perform.
10) sub_2 uses the same operation.

C. Algorithm Pseudo Code

The pseudo code of LoadRate calculation and Load Balancing algorithm are shown in Tables II and III.

IV. Study

A. Tourism System Description

While the standard of living is increasing, the tourism industry is thriving day by day. Web service mobile host has become a major hot spot. Application of these case studies is on load balancing algorithm for Web service mobile host. Deployment of the tourism system on the mobile host is to achieve the division and balance of the request.

The tourist system can provide the tourist information inquiry, the target city inquiry, the weather forecast inquiry, the travel ticket inquiry, the airline ticket reservation, and so on.

B. Case Implementation

1) Client sends the request which includes the tourist spot inquiry, the weather forecast inquiry, the airline ticket inquiry and the airline ticket reservation service. Service request graph is shown in Fig. 2.

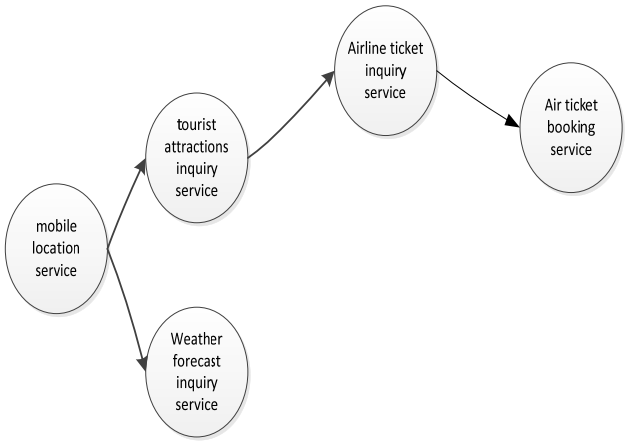

Fig. 2 Service request digraph

2) The Load Balancing scheduler executes the request according to the auxiliary mobile host executive capacity in the resource record form of auxiliary mobile host. According to the Load Salancing algorithm, the auxiliary mobile host looks for the host where resources are the closest ones to the sub-requests needed. The service is then sent to the corresponding auxiliary mobile host. The main code is shown in Table IV.
3) According to the Dinic algorithm, the partition result is shown in Fig. 3.
4) According to the auxiliary mobile host execution ability of the Load Balancing algorithm, the request is divided into two sub-requests with the Dinic algorithm, as shown in Fig. 4.

According to the results of the division, to find the auxiliary mobile host where the resource is closest to the sub-requests, Mobile host sends the sub-requests to the auxiliary mobile hosts and eventually returns the results to the client.

TABLE II
LOADRATE CALCULATION ALGORITHM

Input: Auxiliary mobile host RAM, ROM, Battery, CPU and other information
Output: LoadRate Value of each auxiliary host
1**:** enter auxiliary host RAM, ROM, battery, CPU and other information
2: **while**(i<=n)
3: calculate the percentage of the remaining RAM memory =Pram[i]
4: calculate the percentage of the remaining ROM memory =Prom[i]
5: calculate the percentage of the remaining battery memory = Pbattery[i]
6: calculate the percentage of the remaining cpu memory= Pcpu[i]
7: loadRate[i] = Pram[i] +Prom[i]+Pbattery[i] + Pcpu[i]
8: save all of the auxiliary host LoadRate values in the table
9: **end while**

TABLE III
LOAD BALANCING ALGORITHM

Input: Service request
Output: Request to be sent to an auxiliary mobile host
1: calculate $L_1 \ldots\ldots L_n$, S_{need} , $S_{A1} \ldots\ldots S_{An}$
2: **if** (auxiliary mobile host can execute the request) **{**
3: **if**(only one auxiliary mobile host can execute)**{**
sent the request to the auxiliary mobile host**}**
4: **else{**sent the request to the optimal auxiliary mobile host**}}**
5: **else{**
6: **if** (request is a single-request)**{**return to client cannot execute the request**}**
7: **else**{according to Dinic divide the request into two sub- requests;
8: **if**(only one auxiliary mobile host can execute)**{**
sent the sub_1 to the auxiliary mobile host**}**
9: else{sent the sub_1 to the optimal auxiliary mobile host**}**
10: if($S_{Ax1} + S_{Ax2} >= S_{sub1}$ **){**
11: if(only one auxiliary mobile host can execute the sub_1) {sent the request to the auxiliary mobile host**}**
12: else{sent the request to the optimal auxiliary mobile host**}**
13: else{
14: if($S_{Ax1} + S_{Ax2} + S_{Ax3} >= S_{sub1}$ **){**
15: if(only one auxiliary mobile host can execute the sub_1){ sent the request to the auxiliary mobile host**}**
16: else{sent the request to the optimal auxiliary mobile host**}**
17: else{
18: if($S_{Ax1} + S_{Ax2} + S_{Ax3} + S_{Ax4} >= S_{sub1}$ **){**sent the request to the auxiliary mobile host **}**
19: else{Divide the sub1 into two sub-request by Dinic**}**
}
}
}
}
}
}

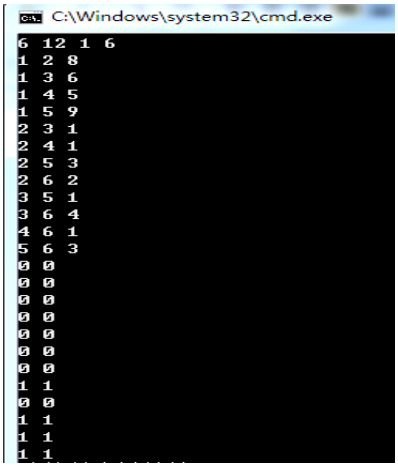

Fig. 3 Dinic algorithm implementation results

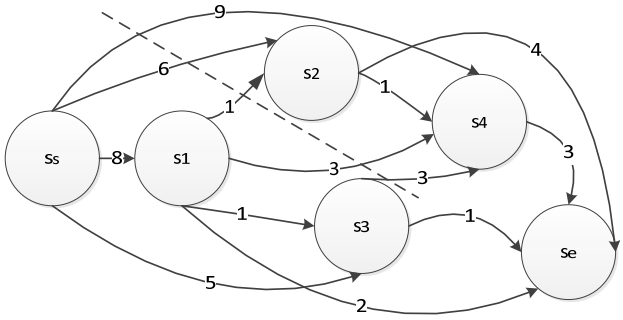

Fig. 4 Dinic algorithm segmentation results

TABLE IV
THE MAIN CODE

```
socket = new Socket(AMHIP, 8888);
DataOutputStreamout=newDataOutputStream(socket.getOutputStream());
out.writeUTF(sendMsg);
out.flush();
DataInputStreamin=newDataInputStream(socket.getInputStream());
String readMsg= in.readUTF();
if(readMsg!= null) {
text.setText(readMsg);
        }

try {
output= socket.getOutputStream();
input = socket.getInputStream();
BufferedReader buffer =newBufferedReader(newInput StreamReader(input));
        output.write(str.getBytes("gbk"));
        output.flush();
        socket.shutdownOutput();
        while ((line = buffer.readLine()) != null) {
                System.out.print(line);
        }
}
```

V. CONCLUSION

This paper studies the Load Balancing algorithm for Web service on the mobile host. Web service and Load Balancing technology on the mobile host will improve the execution ability of the mobile host. This paper implements when the mobile host resource is insufficient, the request is divided into sub-requests, and these sub-requests are sent to the different auxiliary mobile hosts to execute. Finally, the mobile host will integrate the requests and return it to the client. This will shorten the execution time and improve the efficiency.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China (Grant No. 61272172, 51179020, 51479021).

REFERENCES

[1] Asif, Muhammad, Shikhairesh Majumdar. Hosting Web Service on Resource Constrained Devices. International conference on Web Service, 2007,7 :583-590.
[2] Alshawan, Faisal. Fragmentation, Orchestration and Federation of Complex resource-based Mobile Web Service. In IT CoNvergence PRActice, 2014, 3:51-78.
[3] Grondzak, Kortis. Cluster Load Balancing algorithms comparison. Emerging Learning Technologies and Applications, 2013 ,11:225-228.
[4] Qiuxi Zhong, Taoxie Chen. A coevolutionary approach to task allocation and scheduling. Journal of Computer Science, 2001,3:309-313.
[5] Shengyong Huang. Research on MDB Load Balancing and asynchronous concurrent short message distribution system. Modern computer, 2014,26:10-15.
[6] Jian Liu, Xuzhi Li. Research and implementation of a dynamic Load Balancing mechanism. Computer engineering and Applications, 2006,02:142-145.
[7] Asif, Majumdar. Performance Analysis of Mobile Web Service Partitioning Frameworks. Danced Computing and Communications, 2008,16:190-197.
[8] Sathya, Gobi, Kirubakaran. Performance Analysis of Partitioning Mobile Web Services. Emerging Trends in Science, Engineering and Technology *(INCOSET)*, 2012,10:139-142.
[9] Shuixiao Hao, Dandan shen. Dynamic Load Balancing Algorithm in cloud. (UIC-ATC-ScalCom) ,2015,12.
[10] Xuezeng Pan, Kang Sun, Kuijun Lu, Jimin Wang, Ling Di Ping. Dynamic configuration system task time domain partition algorithm. Journal of Zhejiang University, ,2007,11:1839-1844.
[11] Feda AlShahwan, Klsud Moessner. Providing SOAP Web Services and RESTful Web Services from mobile Hosts. Internet and Web Applications and Services (ICIW) 2010,5:174-179.
[12] Kun Yang, Shumao Ou. On Effective OffLoading Services for Resource Constrained Mobile Devices Running Heavies Mobile Internet Applications. IEEE, Communications Magazine 2008,46:56-63.
[13] Kamal Eldin Mohamed, Duminda Wijesekera. A lightweight Framework for Web Services Implementations on Mobile Devices. Proceedings of the 2012 IEEE First International Conference on Mobile Services. 2012: 64-71.
[14] Rahul Garg, Akshat Verma. Balancing Stream Assignment for Service Facility. International Conference on High Performance Computing. 2010,12:1-10.
[15] Mohan, Liyanage. Lightweight Mobile Web Service Provisioning for Sensor Mediation. Proceedings of IEEE International Conference on Mobile Services, 2015:57-64.
[16] Josephine Spinoza Jane, Prof Somasundaram. Discovering and Accessing Web Service in Mobile Devices Using a Load Balancing Proxy-Based Service. Green Computing, Communication and Conservation of Energy (ICGCE), 2013,02:59-64.
[17] Qi Wang. Research on the application of resource constrained Web Service. Dalian Maritime University, 2015.