International Journal of Innovations in Computing
(ISSN : 2319-8257) Vol. 2 Issue 2

Study & Analysis of Mobile Milieu Using Mobile Cloud Computing

Rahul Sharma
Research Scholar
SNU Ranchi (India)
aquahpu@gmail.com

Rohit Sharma
Sr. Consultant
Essar Group, Mumbai (India)
it2k.rohit@gmail.com

Dr. Amod Tiwari
Dept. of CSE
IIT Kanpur(India)
amodtiwari@gmail.com

***Abstract*—Cloud computing has been the most prevalent technology in the past few years and many prominent enterprises have prompted their cloud systems, preparing for the coming age of cloud computing. We can predict that the mobile area will take on a boom with the advent of this new concept. While due to the inherent characteristics of mobile environment, challenges like mobility, heterogeneity and low band-width will hinder the advancement of this incorporation. In this paper we studied & analysis a new concept and an abstraction derived from mobile agent-Universal Mobile Service Cell to shield the un equivalence of heterogeneous distributed systems between mobile devices and the cloud. Then scheduling the cell plays a key role in the mobile cloud computing. Here we studied the Genetic Algorithm to meet this challenge. This paper is aimed at proposing a analytic solution from the architecture to the algorithm.**

Keywords-cloud computing, mobile computing, mobile agent, mobile cloud computing, scheduling, universal mobile service cell, genetic algorithm

I. INTRODUCTION

Cloud computing has been widely regarded as the next generation computing infrastructure. It is a paradigm in which information is permanently stored in servers on the Internet and cached temporarily on clients [10].

Then we can incorporate it into the mobile environment. Although wireless network brings us many benefits, the challenges also emerge to hinder the development of mobile computing.

So we will discuss the challenges of mobile computing first. And then we will study architecture to meet those challenges. The core concept of this architecture is Universal Mobile Service Cell abbreviated as UMSC which derives from mobile agent and also provides an abstraction of mobile agent.

And finally we concentrate on the one of the most basic problem of computing, namely scheduling. Scheduling is the key to implement production with high efficiency, flexibility and reliability. The main purpose of it is to schedule cells to the adaptable nodes on the cloud in accordance with adaptable time, which in fact involves finding out a proper sequence in
which cells can be executed under transaction logic constraint.

Moreover, in order to shield the un equivalence of heterogeneous distributed systems between mobile and the cloud, we should schedule UMSC instead of tasks or processes; as a result, we achieve a higher level of abstraction. Here we choose Genetic Algorithm for scheduling. Though GA is aims at solving combinatorial optimization problem, there must be an oversize solution space to be searched inefficiently. So a future work is needed to improve its efficiency.

The rest of this paper is organized as follows: in section II, we will study optimized mobile architecture which contains three key concepts: mobile cloud, UMSC and mobile hosts. In section III, the Genetic Algorithm is discussed mathematically. In section IV, we will conclude the whole paper and propose a future work for a better solution.

II. OPTIMIZED MOBILE ARCHITECTURE

Along with the advantage brought by wireless network, mobile environment faces more challenges. [1] Firstly, in a wireless environment, there are many obstacles. As a result, mobile communication is characterized by lower bandwidths, high error rates and more frequent spurious disconnections. Secondly, the character of mobile environment, mobility causes the fast address migration and dynamic configuration which makes it more difficult for the system to answer the queries from users efficiently and timely. Thirdly, the portable device in the mobile environment always can't have powerful computing ability and storage ability. That means the mobile host can't support enough computing and storage ability for some complex services which are needed by users.

To solve these problems, this paper contains an optimized mobile architecture. It is a hybrid solution combined mobile-agent technology with mobile cloud computing. The mobile environment is divided into lots of cell regions.[2] Every cell region has several cloud units which act as mobile support station to support services for mobile users in this region. Cloud units in every cell region and remote cloud units are connected to be the mobile cloud, which supports computing ability and storage ability in the way of providing many services. The mobile cloud alleviates the mobile host from complex computing and the mobile hosts can just concentrates on the interaction with users. For the transmission between mobile host and cloud units, we analysis Universal Mobile Service Cell which acts as an abstraction of the mobile agent to solve the influence of unstable network. (Figure 1 shows the architecture.)

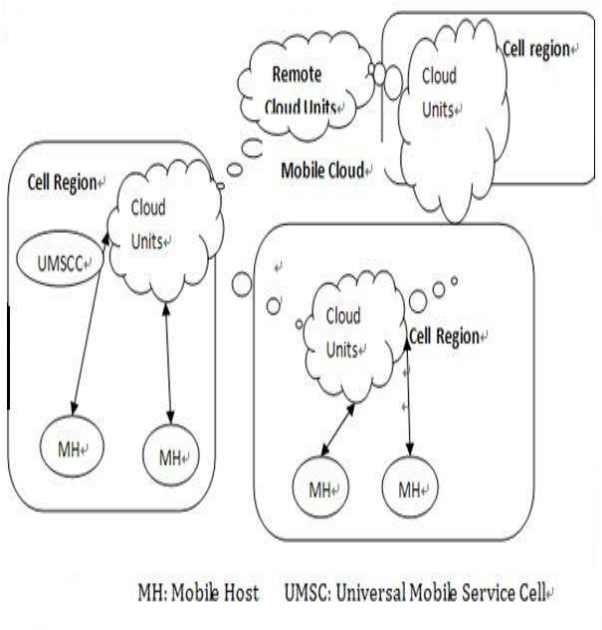

Fig. 1: Architecture

In a word, this architecture has three key aspects: mobile cloud which supports the service, Universal Mobile Service Cell which acts as service agent for users, mobile host. We will explain the architecture in these three aspects.

A. Mobile Cloud

The mobile cloud is an instance of technology using cloud computing in mobile environment. The cloud computing is based on a collection of many old and few new concepts in several research fields like Service-Oriented Architecture(SOA), distributed and grid computing as well as virtualization and allows users to temporary utilize computing infrastructure over the network, supplied as a service by the cloud-provider at possibly one or more levels of abstraction.[3][10] The key opinion expressed by the cloud computing is to transfer the complex computing to the cloud and the service-oriented concept. For the constraint of computing ability of portable device, the cloud computing is so suitable for the mobile environment. So we proposed the mobile cloud to solve this constraint. With mobile cloud, users just need to send their requests for certain service and the cloud provides the service. The mobile host does not need to pay much computing time for complex services. The advantage of cloud computing is brought to the mobile environment with this architecture.

The mobile cloud consists of two kinds of cloud units: cloud units in every cell region and remote cloud units. It has some differences from formal cloud. Firstly, it's asymmetric. These two cloud units have different computing ability and its main task is different. The former one aims at dealing the requests from users directly, the latter one is for dealing the key computing of some service. Cloud units in cell region send requests to the remote cloud units for some complex service. (Figure 2.1 shows the architecture.)

The mobile cloud is divided into five layers in perspective of the compos ability of the system [4].

- Cloud Application Layer
- Cloud Software Environment Layer
- Cloud Software Infrastructure Layer
- Software Kernel
- Hardware and Firmware

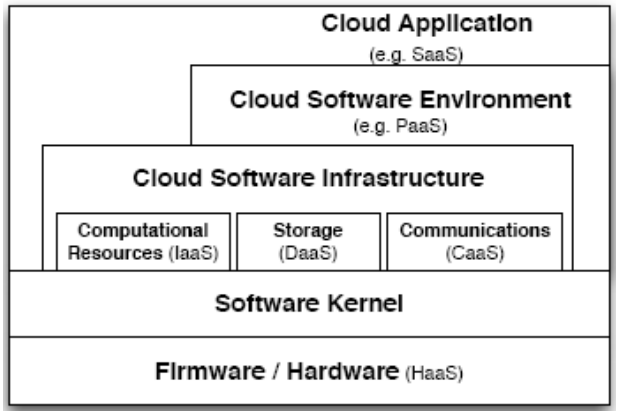

Fig. 2 : The structure and inter-relation between each layer.

B. Universal Mobile Service Cell

Wireless connection is unstable and it influences the quality and stability of service provided by the mobile cloud. To solve this problem, we propose Universal Mobile Service Cell which serves as a proxy for transmission between mobile hosts and the mobile cloud. It is an intelligent software module which brings the request from users and migrates in the mobile cloud to search the response for the request. It has two kinds of cell: User Mobile Service Cell and Cloud Mobile Service Cell. (Figure.3 shows the communication.)

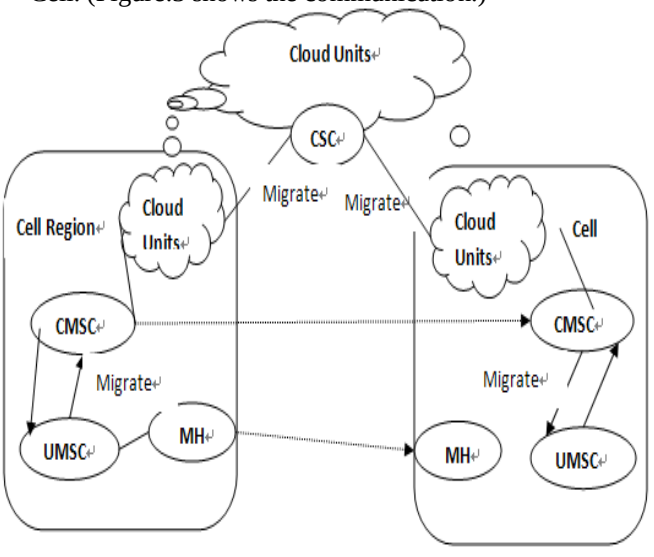

Figure 3: Communication

The detail process of the communication between MH and MC based on Universal Mobile Service Cell is described below [5]. Firstly, in mobile host, the agent called User Mobile Service Cell collects the requests from the user.

Secondly, the agent migrates to the corresponding cloud units and it's called Cloud Mobile Service Cell.

Thirdly, Cloud Mobile Service Cell migrates in the mobile cloud and search for the cloud units which can meet the requests. Then it brings the result with itself and migrates to the corresponding MH.

Universal Mobile Service Cell do not need of sending the request/response messages with low bandwidth wireless connection between the mobile cloud and mobile host. As service cell itself migrates, this problem can be solved with no continuous communication with the mobile host. Then the challenge that comes from sudden disconnection of wireless network and the situation of turning mobile host off for power saving is transparent to users.

Two problems comes up in this case: what can this system react for the situation when the mobile host is disconnected while cloud-service-cell wants to return the result from the user's request to mobile host , or when the mobile host get out of this cell region into another cell region. For the first problem, cloud units in certain cell region keep a
list of connection information between every MH and MB.

When Cloud Mobile Service Cell finishes the searching work, cloud units check the list to judge if the connection is kept. If it is connected, the formal workflow is done. If it is not connected, cloud units use a proxy system to save the certain Cloud Mobile Service Cell until the connection is rebuilt. For the second problem, using the IP Mobility Support, cloud units in certain cell regions can know the change of the mobile host. The Cloud Mobile Service Cell migrates to the corresponding cloud units and repeats the above process.

The Universal Mobile Service Cell makes the mobility transparent to users and also minimizes the bad influence of the unstable wireless network. It is like an intelligent bridge connecting MH to MB.

C. Mobile Host

The Von Neumann's Architecture [6] has dominated computer systems for many years. However, firstly, the separation between the CPU and memory leads to the von Neumann bottleneck, the limited throughput (data transfer rate) between the CPU and memory compared to the amount of memory. So with the advent of cloud computing and the great improvement in the hardware, we can improve this architecture reasonably. Secondly, as it is mentioned above, the portable device can't have powerful computing ability and storage ability. The cloud computing provides us with unlimited resources, great capacity for storage and the computing ability theoretically. We can combine the CPU and the memory to one component named ***CM***, moreover an important new component called **Mobile *Cloud*** is attached to the new architecture.

In the perspective of users and mobile host, the host computer architecture is illustrated below. (Figure 4) The local host just deals some easy tasks and concentrates on the interaction with users, leaving the complex computing to the mobile cloud.

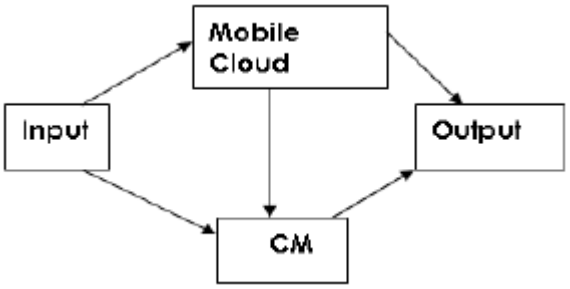

Fig. 4 : Host computer architecture

III. ALGORITHM

A. Background

Mobile service scheduling is the core problem of the mobile cloud computing. Traditionally, the scheduling methods are divided into two categories, namely heuristic and meta-heuristic. List heuristics assign each task a priority and sort them in decreasing order. Most of them are efficient but often can't obtain reasonable solutions in all situations.

The meta-heuristic method, famous as Genetic Algorithm, is a guided random search method which mimics the principles of evolution and natural genetics. Because genetic algorithms search optimal solutions from entire solution space, they often can obtain reasonable solutions in all situations.

The scheduling on the mobile cloud computing environment faces the challenge that the heterogeneous distributed systems are un equivalent [9], there are a huge difference concerning about the capability of computing of mobile device and the cloud. The solution is to provide an abstraction over the heterogeneous environment.

In this section we will analysis a concept to shield the unequivalent of the heterogeneous systems and study an algorithm based on GA [7] [8] for scheduling in the mobile cloud computing environment.

B. Model and definitions

Every mobile cloud service is composed of many cells; we call it Universal Mobile Service Cell, as mentioned in above section. Due to its mobility in the mobile environment, we divide it into two categories, User Mobile Service Cell and Cloud Mobile Service Cell. The concept of cell is very similar to the mobile agent, or in other word, it is an abstraction over the mobile agent. So it is inherent to obtain some characteristics.

1) ***Definition 1***. UMS (Universal Mobile Service) is a set of cells; Cell is a triple that is Cell (arrival time, priority, state). Cell->state $\in$ {migrated, divided}, the state demonstrate whether the cell is migrating or divided to different nodes.

UMS = {C_1, C_2, C_3...Cn | n $\in$ N+}, so scheduling the service of the cloud is scheduling the cells which consist of the service, however there are some rules

that the scheduling must satisfy. The constraints are as follows:

- *Precedence:* if C_i must be completed before C_j can be initialized, then we denote this precedence relation as $C_i < C_j$, the precedence can also be denoted as a DAG (Directed Acyclic Graph). The precedence is determined by the arrival time, priority and computational time of the cell.
- *Integrality:* C_1, C_2, …C_n, the n cells combine together to complete a service no matter what precedence they may form.
- *Uniqueness: The precedence of the cells identifies the service.*

2) ***Definition 2***. **GASS** (Genetic Algorithm Scheduling System) is a five-tuple $\prod (N, R, C, D, G) \subseteq N\times R\times C\times D\times G$

- **N** is a set of the cloud nodes which perform the computation. **$N= \{N_1, N_2, N_3 \ldots N_m\}$** our goal is the map the UMS to N to obtain the least computational time for the service. Mathematically, we denote it as : **Solution: UMS → N**
- **$R= \{r (N_i, N_j) \mid r (N_i, N_j) \subseteq N\times N\}$** it is a matrix which represents the migrating time between the cloud nodes Ni and Nj. $r (N_i, N_i) = 0$.
- **$C= \{c (C_i, N_j) \mid c (C_i, N_j) \subseteq UMS\times N\}$** it is a matrix which represents the computational time when cell Ci is executed in cloud node Nj
- **$D = \{d (C_i, C_j) \mid d (C_i, C_j) \subseteq UMS\times UMS\}$** it is a matrix which represents the transmitting time of the data between cell C_i and C_j. $d (C_i, C_i) = 0$.
- **G = (C, E)** it is the DAG which is used to denote the precedence of the cells; **E** represents the edges between the cells in the DAG.

.

C. The Genetic Algorithm on the cloud

1) Genetic Representation

The individual genetic representation is a solution for the scheduling problem. Here we define it as Genetic $(C_i, N_j) \subseteq C\times N$, such as the solution in figure 4:

C_1	C_2	C_3	C_4	C_5	C_6
N_1	N_2	N_3	N_1	N_3	N_2

Fig. 4 : Solution

Then the individual solution set will be Solution= {Genetic (C_i, N_j) | Genetic $(C_i, N_j) \subseteq C\times N$}. Then

Solution={Genetic(C_1,N_1),Genetic(C_2,N_2)…Genetic (C_6, N_2)}

2) Initial Population

The Population is the set of solutions, traditionally, we randomly initialize the Population, while due to the precedence of the cells and the efficiency of the computing,
here we choose the better way to initialize it.

First, if C_i is the node of DAG, we define that if pred $(C_i) = \phi$, C_i is the In-Node of the DAG opposite to the Out- Node. Then the initializing algorithm (we will use the symbols defined above) will be:

```
for  i to (population size)
    while T ≠ϕ
                Ci= random (UMS)
                Nk= random (N)
          if pred (Ci) =ϕ
                E= E- {(Ci, Cj) | (Ci, Cj) ∈ E  ∧ Ci < Cj }
                C= C-{Ci}
                UMS=UMS-{Ci}
                Solution->add(Genetic(Ci, Nk))
           end if
       end while
                Population->add (Solution)
 end for
```

comment: E is the edge set of the DAG (here we name it G); C is the node set of G.

3) Fitness Function

The Fitness Function in Genetic Algorithm is typically the function that we want to optimize, here we want to minimize the computational time of the scheduling. So the Fitness Function will be:

F (Solution) = C + D + R

C, D and R represent the corresponding element in the matrix according to the Solution.

While we always want to maximize the value of the fitness function, so we change it into:

Fitness (Solution)= S－F(Solution)

S is a constant which assures that Fitness (Solution) >0

4) Crossover

Crossover means that two possible Solutions form a new Solution via the GA operators. We will use the following method:

- select sites where we can cut the genetic representations into two halves. We assume that there are two legal solutions S_1, S_2. We randomly choose a cell P named crossover-cell whose position is P_1 in S_1 and P_2 in S_2. Then both S_1 and S_2 are divided into left and right parts.
  Mathematically we define the process as
  $S_1= SL_1 \cup SR_1$, $S_2= SL_2 \cup SR_2$.
- Then the exchange begins. We choose a mathematical way to denote it as:

$$S_1^* = SL_1^* \cup SR_1^*, S_2^* = SL_2^* \cup SR_2^*$$

SL_1	1	2	1	2
SL_2	1	2	2	1

We will prove that the new solution satisfies these constraints: precedence, integrality and uniqueness.

It's easy to prove the integrality and uniqueness. Then we will focus on the precedence.

Proof:

$\because$ $\forall$ $cell_i \in SL_1, SL_2$
$celli < P$
$\forall$ $cell_i \in SR_1, SR_2$
$P < celli$

$\therefore$ $\forall$ $cell_i \in SL_1 \cup SL_2$
$celli < P$
$\forall$ $cell_i \in SR_2$
$P < celli$

$SL_1^* = SL_1 \cup SL_2, SR_1^* = SR_2$

$\therefore$ S_1^* satisfies the precedence constraints and so does S_2^*

5) Mutation

Mutation can be considered as an occasional random alternation of the genetic representation. One can take it as an escape mechanism for premature convergence. As for the scheduling, mutation is applied by randomly exchanging two genetic representations with the same cell precedence.

We will use the following method:

- Randomly choose a genetic representations G_i.
- Search a genetic representation G_j having the same precedence with G_i.
- Exchange them.

6) The architecture of the algorithm

Now we can combine all the individual algorithms discussed above to form the GASS on the cloud.

```
Initialize ()
Compute the fitness values
Repeat
        Crossover ()
        Mutation ()
        Compute the fitness values
        Preserve the best string
Util convergent
```

The pseudo code above searches the solution space to check whether the solution fit the terminating condition and finally obtain the best one.

IV. CONCLUSION

In this paper, we discuss an abstraction over the mobile agent and analysis a new way for scheduling in the mobile cloud environment. The GA is analyzed to solve this NP problem. However more efforts are needed to improve the efficiency of the algorithm. The challenges focus on the crossover and mutation and how they can generate a better genetic representation.

REFERENCES

[1] George H. Forman and John Zahorjan University of Washington "The Challenges of Mobile Computing", COMPUTING MILIEUX, April 1994.

[2] Phyoung Jung Kim and Young Ju Noh, "Mobile Agent System Architecture for supporting Mobile Market Application Service in Mobile Computing Environment" in proceedings of IEEE conference on Geometric Modeling and Graphics, July 2003.

[3] "Cloud computing - Issues, research and implementations", ITI 2008, 30th International Conference on Information Technology Interfaces, June 23, 2008 - June 26, 2008

[4] Lamia Youseff, Maria Butrico and Dilma Da Silva, "Toward a Unified Ontology of Cloud Computing", 2007

[5] R. Punithavathi, Dr. K. Duraiswamy "An Optimized Solution for Mobile Computing Environment", International Conference on Computing, Communication and Networking, 2008

[6] First draft of a report on the EDVAC Source Technical Report，1945，John von Newmann

[7] Edwin S H，Ansari N. Genetic Algorithm for Multiprocessor Scheduling [j]. IEEE Trans. on Parallel and Distributed Systems，1994, (52)

[8] "Job-shop scheduling using genetic algorithm" Wu Ying; Li Bin; Systems, Man, and Cybernetics, 1996, IEEE International Conference on Volume 3, 14-17 Oct.1996 Page(s):1994 - 1999 vol.3Digital Object Identifier 10.1109/ICSMC.1996.565434

[9] Heterogeneous distributed genetic algorithms based on the crossover operator Herrera, F.; Lozano, M.; Genetic Algorithms in Engineering Systems: Innovations and Applications, 1997. GALESIA 97. Second International Conference On (Conf. Publ. No. 446)2-4 Sept. 1997 Page(s):203 - 208

[10] " Above the Clouds: A Berkeley View of Cloud Computing" Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz,Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia, February 10, 2009

[11] "Scheduling multi-task multi-agent systems", Proceedings of the fifth international conference on Autonomous agents table of contents Montreal, Quebec, Canada, 2001

[12] Improved genetic algorithm for scheduling divisible data grid application", Abduh, M.; Othman, M.; Ibrahim,H.; Subramaniam, S.;, Telecommunications and Malaysia International Conference on Communications, 2007. ICT-MICC 2007. IEEE International Conference on 14-17 May 2007 Page(s):461 - 465

[13] Analysis of Scheduling Techniques in Distributed Parallel Environments Using Mobile Agents", Marques, G.M.; Sabatine, R.J.; Castelo Branco, K.R.L.J., Networking and Services, 2008. ICNS 2008. Fourth International Conference on 16-21 March 2008