
Resource Optimization for Cloud Environment using Fuzzy Bee Colony Technique

K.Govinda^{#1}, Yuvaraj kumar^{#2}

#1 SCSE,VIT University,8925467665

#2 SCSE,VIT University,9940930098

ABSTRACT

Cloud Computing is considered to be the revolting concept of today's computing technology and has given an enormous attention around the world. Cloud provides numerous number of resources which were on demand in a pay per use method. Cloud computing today had enhanced itself to take its resources to the end user anytime anywhere. That is why large public and private organizations, enterprises have started to discover the significance of the Cloud Computing concept through the documented successes of Amazon and others. In the Cloud provider's perspective with the large number of resources available It becomes hard to manage those resources and moreover accessing and delivering those resources becomes a challenging task. In order to overcome these issues resources optimization in cloud environment becomes essential. Hence in this paper we present a prototype through which the cloud resources can be optimized using "Fuzzy Bee Colony Technique".

Key words: Cloud Computing, Scheduling, Fuzzy Logic, Waggle dance, Hive.

Corresponding Author: K.Govinda

INTRODUCTION

Similar to the day to day utility services like water, gas, electricity and etc. The trend of cloud computing had modernized itself to provide a large number of resources to the end user. The resources include different computational services of different quality, tax calculation service, weather information service, shipping status service etc. The concept cloud computing aims to be global and provides such computational web services to the masses, reaching to the end user that hosts their personal documents on the internet to enterprises outsourcing their entire IT infrastructure to extrinsic data centres. In fact, cloud has given a new approach to make IT a real utility which is global and complete for the end users. It is difficult to define the cloud computing. Cloud Computing is a virtual pool of computing resources. It provides computing resources in the pool for users through internet. Integrated cloud computing is a whole dynamic computing system. It provides a mandatory application program environment. It can deploy, allocate or reallocate computing resource dynamically and monitor the usage of resources at all times. Generally speaking cloud computing has a distributed foundation establishment, and monitor the distributed system, to achieve the purpose of efficient use of the system. Thus the cloud computing trend allows the user to perform some application specific operation on their own digital asset. Some of the cloud resources allows the user to customize the service based on the user's convenience. Programs built using this model will run across multiple platforms

extracting information from each of them combining and delivering it in their own form to any device anywhere in the planet. The power of services offered by the cloud environment is unlimited and these web services are randomly scattered over different servers and accessing and scheduling them which is really a challenging task. For this, we have used honey bee colony algorithm. In nature, we find a lot of interesting collective behaviour of ants, bees, etc. It is observed that honey bees work in a localized, self motivated manner. The honey bee colonies practice division of labour between the forgers, who work in the field collecting the nectar and the food-stores who work in the hive to store nectar. The forger bees search the nectar at random and after getting specific nectar they return back to the hive and start waggles dance as shown in Fig1[4]. In order to motivate the follower forgers to access that very nectar. The duration of this dance is closely correlated with the search time experienced by the dancing bees.

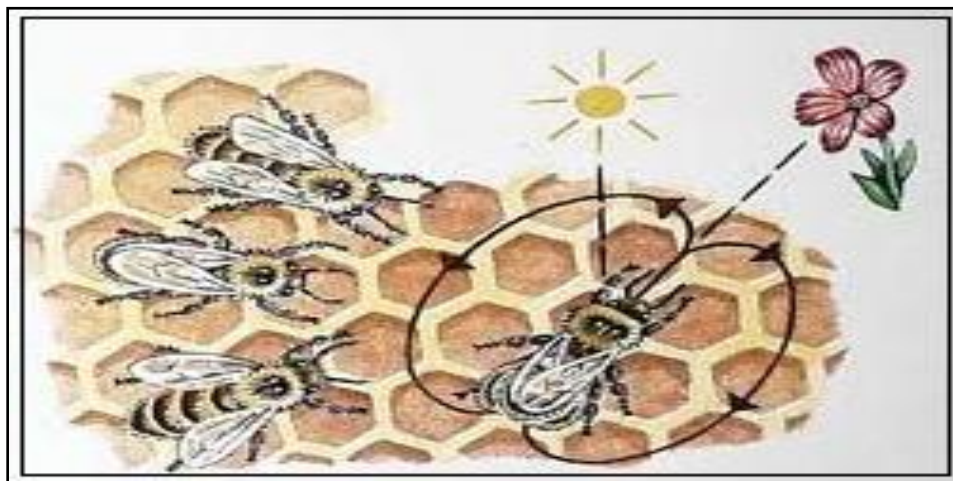


Fig1. Bees Waggle Dance

MATERIALS AND METHODS

Bee Colony Optimization

From the beginning of the world, many natural systems of the most creatures in the world are very rich topics for the scientific researchers. However, a simple idiosyncratic way of approaching a problem in the biodiversity can cooperate to propose a procedure a system able to solve a real complex problem and perform very well-informed tasks. In reality there are many patterns form such protocol like ant colonies, bird flocking, fish shingling, animal herding, bacterial growth, bee colonies, and human neuron system and etc...The study of these kind of procedures and the research on them leads to the growth of the field known as “Artificial Intelligence”.

Artificial Intelligence (AI) is a concept of study and research for finding relationships between cognitive science and computation theories and implementing them. These relationships after some research and implementation are represented as data structures, search techniques, problem solving methods, or representation forms for knowledge. These systems are working either under oversight or without supervision with a strong programmed background. Most of social insects work without supervision with the help of artificial intelligence. By the way, their

teamwork is largely self-organized, and coordination arises from the different levels and methods among individuals in the system that leads to the required outcome. These interactions might be primarily, sounds like ants follow odour trails, or more complex, like a honey bee dancing. The collective behaviour that emerges from a group of the social insects has been enhanced by artificially represented as a technique known as Swarm Intelligence [7].

The Behaviour of Scout Scenario

Scouts fly around the area in search for nectar . When they find a source of nectar or pollen they fly back to their home which is the colony and start dancing to communicate with other bees on a particular region in the comb. Hence the behaviour of the scout scenario is summarized according to the following activities.

The scout flies from its colony searching for food sources in a random way. Once it completes a full trip around the area, it returns back to its colony. When a scout arrives at the colony, it goes inside and announces its presence by vibrating the wing which denotes that it found a place where the nectar is available. If it has found a nearby source of nectar or pollen, it undergoes a circular dance. The nearby bees follow it through the circular dance and smell it for the identification of the source. The intensity of the vibrations in the wing is to pinpoint the coordinate values of the food source. If the source is so close, no directions are given. Alternatively, if the flower source is a long ways off, careful directions will be given. The abstract convention that the scout makes is that the top position on the comb is the position of the sun. Because bees can see polarized light, they can really estimate the sun's position without actually seeing the sun. The scout dances in a precise angle from the vertical. This equals to the horizontal angle of the sun with reference to the colony exit with the location of the food source. Next the scout bee must intimate the other bees how far the source is ? This is done by wagging the abdomen from side to side. The slower the wagging, the further away is the distance of the food flower. Thus the dance of scouts points to the direction, distance, and quality of food source as shown in Fig2.

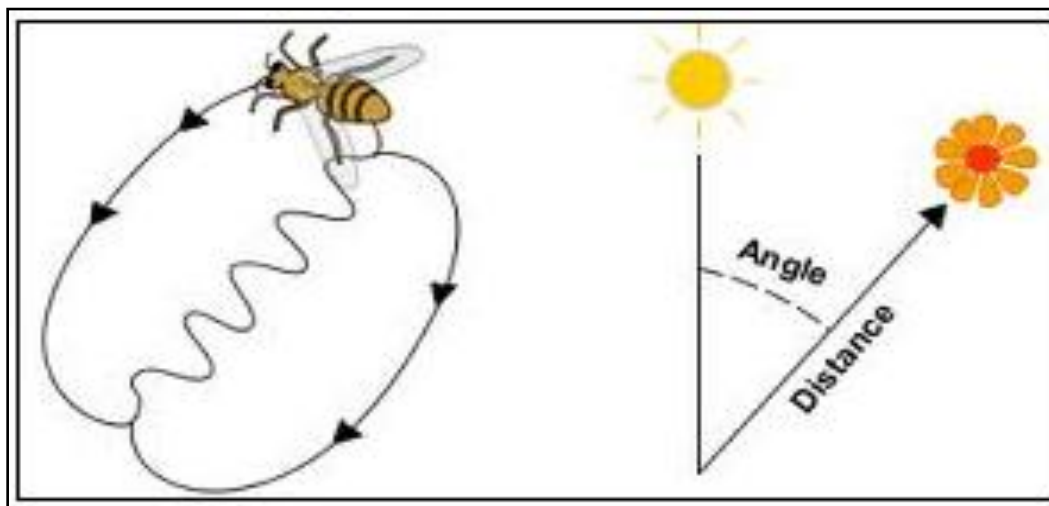


Fig2.Waggle Shows Angle and Distance

The Behaviour of Forager Scenario

The reaction of the forager bees on this show concludes into steps.

The bees in the colony keenly listens the scout bees to learn those directions, and also learn the odour of the flower on scout bee, so they can find the flower when they arrive at the source location. Because the sun is moving in the sky, the bees should use an accurate clock sense to adjust for the changing sun position with reference to the food source and the colony exit. Even more remarkable, if a trained bee is removed from the colony to another location where the flower is not visible, but the colony is, the bee does not return to the colony to get its bearing, but reads sun position, and triangulates, and flies directly to the flower. Subsequently, the forager bees take a load of nectar from the source and return to the colony and unload the nectar to the store of food. Foraging requires energy and the honeybee's evaluation as to where, what, and how long to forage are all related to the economics of energy consumption and the net gain of food to the colony. Generally bees fly only as far as necessary to secure an acceptable food source from which there is a net gain. Therefore, these are the factors that influence foraging behaviour and determine profitability. The net rate of energy intake is defined as the energy gained while foraging minus the energy spent on foraging, divided by time spent foraging.

Mathematical Model

The probability of selecting a nectar source is derived through

$$P_i = \frac{F(\theta_i)}{\sum_{k=1}^S F(\theta_k)}$$

P_i : The probability of selecting the i^{th} employed bee

S : The number of employed bees

θ_i : The position of the i^{th} employed bee

The fitness value is given by

$$F(\theta_i)$$

Calculation of the new position is given by

$$x_{ij}(t+1) = \theta_{ij}(t) + \varphi(\theta_{ij}(t) - \theta_{kj}(t))$$

- x_i : The position of the onlooker bee.

- T : The iteration number and t is time.

θ_k : The randomly chosen employed bee.

- j : The dimension of the solution

$\varphi()$: A series of random variable in the range $[-1, 1]$

The movement of the scout bees follows equation $\theta_{ij} = \theta_{jmin} + r \cdot (\theta_{jmax} - \theta_{jmin})$

$$r \in [0, 1]$$

- r : A random number and

The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using a simple programming model. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures. Technically, Hadoop consists of two key services: reliable data storage using the Hadoop Distributed File System (HDFS) and high-performance parallel data processing using a technique called MapReduce[8]. Originally developed and employed by dominant Web companies like Yahoo and Facebook, Hadoop is now widely used in finance, technology, telecom, media and entertainment, government, research institutions and other markets with significant data. With Hadoop, enterprises can easily explore complex data using custom analyses tailored to their information and questions.

Architecture

Hadoop consists of the *Hadoop Common*, which provides access to the filesystems supported by Hadoop. The Hadoop Common package contains the necessary JAR files and scripts needed to start Hadoop. The package also provides source code, documentation, and a contribution section which includes projects from the Hadoop Community. A Hadoop cluster will include a single master and multiple worker nodes. The master node consists of a JobTracker, TaskTracker, NameNode, and DataNode. A slave or *worker node* acts as both a DataNode and TaskTracker, though it is possible to have data-only worker nodes, and compute-only worker nodes; these are normally only used in non-standard applications. Hadoop requires

JRE 1.6 or higher. HDFS (Hadoop distributed file System) is comprised of interconnected clusters of nodes where files and directories reside. An HDFS cluster consists of a single node, known as a NameNode, that manages the file system namespace and regulates client access to files. In addition, data nodes (DataNodes) store data as blocks within files.

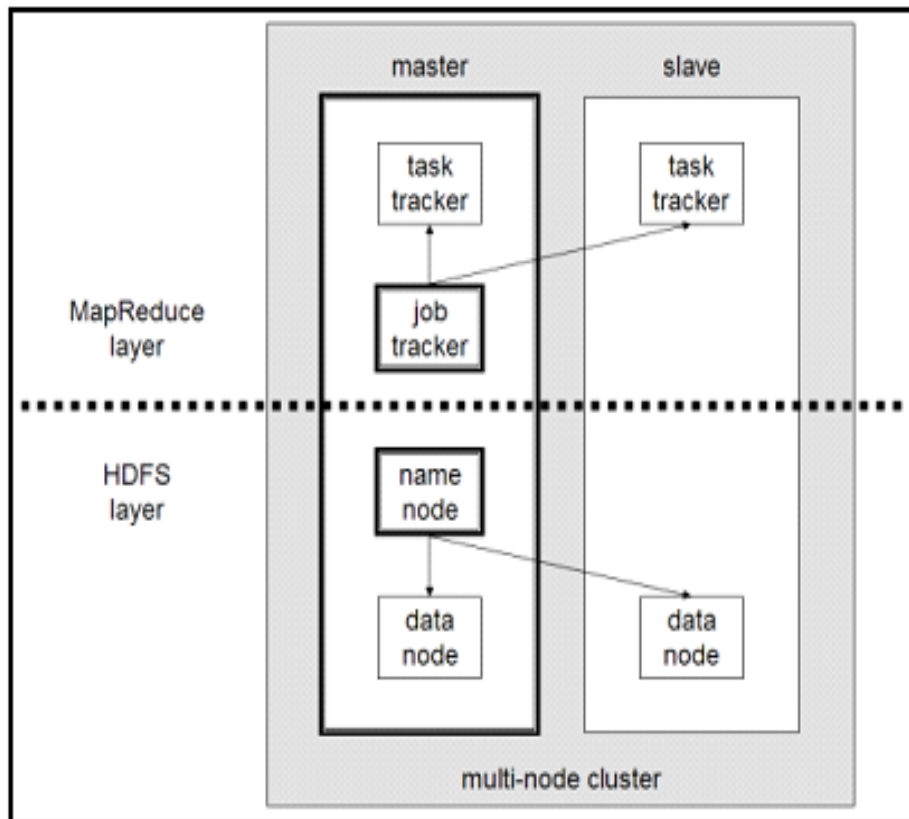


Fig3. Hadoop Architecture

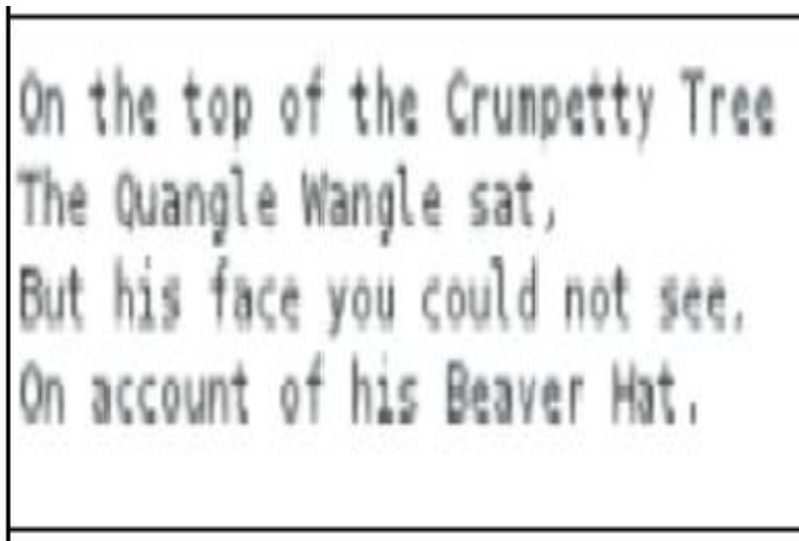
HDFS has a master/slave architecture. An HDFS cluster consists of a single NameNode, a master server that manages the file system namespace and regulates access to files by clients. In addition, there are a number of DataNodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on. HDFS exposes a file system namespace and allows user data to be stored in files. Internally, a file is split into one or more blocks and these blocks are stored in a set of DataNodes. The NameNode executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to DataNodes. The DataNodes are responsible for serving read and write requests from the file system's clients. The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.

MapReduce is a framework for processing highly distributable problems across huge datasets using a large number of computers (nodes)[9-10]. Computational processing can occur on data stored either in a filesystem (unstructured) or in a database (structured). **"Map" step:**

The master node takes the input, partitions it up into smaller sub-problems, and distributes them to worker nodes. A worker node may do this again in turn, leading to a multi-level tree structure. The worker node processes the smaller problem, and passes the answer back to its master node. **"Reduce" step:** The master node then collects the answers to all the sub-problems and combines them in some way to form the output – the answer to the problem it was originally trying to solve. Above the file systems comes the MapReduce engine, which consists of one JobTracker, to which client applications submit MapReduce jobs[11]. The JobTracker pushes work out to available TaskTracker nodes in the cluster, striving to keep the work as close to the data as possible. With a rack-aware filesystem, the JobTracker knows which node contains the data, and which other machines are nearby. If the work cannot be hosted on the actual node where the data resides, priority is given to nodes in the same rack. This reduces network traffic on the main backbone network. If a TaskTracker fails or times out, that part of the job is rescheduled. The TaskTracker on each node spawns off a separate Java Virtual Machine process to prevent the TaskTracker itself from failing if the running job crashes the JVM. A heartbeat is sent from the TaskTracker to the JobTracker every few minutes to check its status.

Implementation

We implemented the same natural bees optimization in HDFS on linux platform by taking the following input from the user and the result as shown in Fig4.



On the top of the Crunpetty Tree
The Quangle Wangle sat,
But his face you could not see,
On account of his Beaver Hat.

Fig4. Input

MapReduce process the input and produces the word count.

```
6715 Child
[enlightened@geek-world ~]$ hadoop jar hadoop-0.20.203.0/hadoop-examples-0.20.203.0.jar wordcount test.txt out6
11/11/17 16:31:17 INFO input.FileInputFormat: Total input paths to process : 1
11/11/17 16:31:17 INFO mapred.JobClient: Running job: job_201111171627_0004
11/11/17 16:31:18 INFO mapred.JobClient: map 0% reduce 0%
11/11/17 16:31:33 INFO mapred.JobClient: map 100% reduce 0%
11/11/17 16:31:45 INFO mapred.JobClient: map 100% reduce 100%
11/11/17 16:31:50 INFO mapred.JobClient: Job complete: job_201111171627_0004
11/11/17 16:31:50 INFO mapred.JobClient: Counters: 25
11/11/17 16:31:50 INFO mapred.JobClient: Job Counters
11/11/17 16:31:50 INFO mapred.JobClient: Launched reduce tasks=1
11/11/17 16:31:50 INFO mapred.JobClient: SLOTS_MILLIS_MAPS=13347
11/11/17 16:31:50 INFO mapred.JobClient: Total time spent by all reduces waiting after reserving slots (ms)=0
11/11/17 16:31:50 INFO mapred.JobClient: Total time spent by all maps waiting after reserving slots (ms)=0
11/11/17 16:31:50 INFO mapred.JobClient: Launched map tasks=1
11/11/17 16:31:50 INFO mapred.JobClient: Data-local map tasks=1
11/11/17 16:31:50 INFO mapred.JobClient: SLOTS_MILLIS_REDUCE=10488
11/11/17 16:31:50 INFO mapred.JobClient: File Output Format Counters
11/11/17 16:31:50 INFO mapred.JobClient: Bytes Written=145
11/11/17 16:31:50 INFO mapred.JobClient: FileSystemCounters
11/11/17 16:31:50 INFO mapred.JobClient: FILE_BYTES_READ=231
11/11/17 16:31:50 INFO mapred.JobClient: HDFS_BYTES_READ=226
11/11/17 16:31:50 INFO mapred.JobClient: FILE_BYTES_WRITTEN=42827
11/11/17 16:31:50 INFO mapred.JobClient: HDFS_BYTES_WRITTEN=145
11/11/17 16:31:50 INFO mapred.JobClient: File Input Format Counters
11/11/17 16:31:50 INFO mapred.JobClient: Bytes Read=119
11/11/17 16:31:50 INFO mapred.JobClient: Map-Reduce Framework
11/11/17 16:31:50 INFO mapred.JobClient: Reduce input groups=20
11/11/17 16:31:50 INFO mapred.JobClient: Map output materialized bytes=231
11/11/17 16:31:50 INFO mapred.JobClient: Combine output records=20
11/11/17 16:31:50 INFO mapred.JobClient: Map input records=4
11/11/17 16:31:50 INFO mapred.JobClient: Reduce shuffle bytes=0
11/11/17 16:31:50 INFO mapred.JobClient: Reduce output records=20
11/11/17 16:31:50 INFO mapred.JobClient: Spilled Records=40
11/11/17 16:31:50 INFO mapred.JobClient: Map output bytes=215
11/11/17 16:31:50 INFO mapred.JobClient: Combine input records=24
11/11/17 16:31:50 INFO mapred.JobClient: Map output records=24
11/11/17 16:31:50 INFO mapred.JobClient: SPLIT_RAW_BYTES=107
11/11/17 16:31:50 INFO mapred.JobClient: Reduce input records=20
[enlightened@geek-world ~]$ hadoop fs -text out5/part-r-00000
Beaver 1
But 1
Crumpetty 1
Hat. 1
On 2
Quangle 1
The 1
Tree 1
Wangle 1
account 1
could 1
face 1
his 2
not 1
of 2
sat, 1
see, 1
the 2
top 1
you 1
[enlightened@geek-world ~]$
```

Conclusion

This paper proposes an algorithm for optimization of resources over cloud using bee colony system according to the two scenarios of scouting and forging processes. The same process can be improved by adding semantics to data over cloud, where the governance of resource becomes highly automated with these type of optimization techniques.

REFERENCE

- [1] Amazon Inc. Elastic Compute Cloud [URL]. Amazon, Nov. 2008.
<http://aws.amazon.com/ec2>.
- [2] Chris Anderson. “*The Long Tail: How Endless Choice Is Creating Unlimited Demand*”. Random House Business Books, July 2006.
- [3] David P. Anderson and Gilles Fedak. “The computational and storage potential of volunteer Computing”. In *CCGRID '06*, IEEE Computer Society, 2006, pages 73–80.
- [4] J. C. Biesmeijer and T. D. Seeley, “The use of waggle dance information by honey bees throughout their foraging careers,” *Behavioral Ecology and Sociobiology*, vol. 59, no. 1, 2005, pp. 133-142.
- [5] L. P. Wong, M. Y. H. Low, and C. S. Chong, “Bee colony optimization with local search for traveling salesman Problem,” in *Proc. of 6th IEEE International Conference on Industrial Informatics (INDIN 2008)*. IEEE, 2008, pp.1019–1025.
- [6] F. C. Dyer, “The biology of the dance language,” *Annual Review of Entomology*, volume.47, 2002, pp. 917-949.
- [7] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence*, Oxford University Press, Oxford, 1999.
- [8] Apache Hadoop. <http://hadoop.apache.org/>.
- [9] J. Dean, S. Ghemawat, “MapReduce: Simplified Data Processing on Large Clusters,” In *Proc of the 6th Symposium on Operating Systems Design and Implementation*, San Francisco CA, Dec. 2004.
- [10] A. Gates, O. Natkovich, S. Chopra, P. Kamath, S. Narayanam, C. Olston, B. Reed, S.Srinivasan, U. Srivastava. “Building a High-Level Dataflow System on top of MapReduce: The Pig Experience,” In *Proc of Very Large Data Bases*, vol 2 no. 2, 2009, pp. 1414–1425.
- [11] S. Ghemawat, H. Gobioff, S.Leung. “The Google file system,” In *Proc. of ACM Symposium on Operating Systems Principles*, Lake George, NY, Oct 2003, pp 29–43.