Volume : 3 | Issue : 2 | February 2014 • ISSN No 2277 - 8179

Research Paper

Achievement of Cost-Effective Datasets Transference in The Cloud Through Ant Colony Optimization

Engineering

KEYWORDS : Ant Colony Optimization, data sets storage, pay-as-you-go model, apriori algorithm,cloud computing

THAHASSIN C	M.E C.S.E,NEHRU INSTITUTE OF TECHNOLOGY,COIMBATORE-641105,INDIA
A. GEETHA	ASSISTANT PROFESSOR,DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING,NEHRU INSTITUTE OF TECHNOLOGY,COIMBATORE-641105,INDIA
RASEEK C	ASSISTANT PROFESSOR OF CSE DEPT,GOVT. ENGINEERING COLLEGE , SREEKRISHNAPURAM,PALAKKAD-679513

ABSTRACT

The booming volume of data generated by contemporary business users and consumers have created enormous data storage and management challenges. Cloud computing provides a way to enable massive amounts of data to work together as data-intensive services.Many users are moving their data to online storage clouds, where data are stored based on the pay-as-you-go model. There are many cost-effective approaches have been developed for achieving minimum cost benchmark. However,it may not be sufficient, if large-scale applications have to run in a more distributed manner. In this paper, for incorporating data transfer cost into minimum cost benchmark, an effective Ant Colony Optimization (ACO) algorithm has been proposed. ACO reduces the delay in accessing & transferring the data sets which paves the way to achieve a minimum cost benchmark

1.INTRODUCTION

In recent years, the widespread generation, transmission and storage of ever increasing types and volumes of data increases the load on the networking and storage infrastructure.Cloud computing is emerging as the latest distributed computing paradigm, which provides reliable, inexpensive, redundant and scalable resources on demand to system requirements.By the IaaS service of cloud, the heterogeneity of computing systems of one service provider can be well shielded by the virtualization technologies. Therefore, users can implement their applications in unified resources without any infrastructure investment, where excessive processing power and storage can be obtained from commercial cloud service providers. The cloud computing services are used on the basis of pay-as-you-go model,where the users have to pay for the storage as well as computation of the data sets & utility services. With this model, the total application cost in the cloud highly depends on the strategy of storing the application data sets.However, storing all the generated application data sets in the cloud may result in a high storage cost, because some large data sets may be infrequently used.At the same time, deleting all the generated data sets and regenerating them every time when they are needed may result in a high computation cost. So there should maintain a balance between storing the frequently used data sets & regenerating the rest when needed.

There are many approaches used for attaining minimum cost benchmark like Cost Transitive Tournament Shortest Path (CTT-SP)-based algorithm which was used for static on-demand minimum cost benchmarking of data sets storage in the cloud. Furhter it has enhanced & deployed a highly cost effective and practical storage strategy which can automatically decide whether a generated data set should be stored or not at runtime in the cloud. The main aim of this strategy is the local-optimization for the tradeoff between computation and storage. It also takes users' preferences on storage into consideration. However, sometimes large-scale applications have to run in a more distributed manner because some application data may be distributed with fixed locations.

In this paper,the data transfer costs are also reduced inorder to maintain an overall minimum cost benchmark through a methodology named Ant Colony Optimization (ACO) with Apriori algorithm. In previous works, the researchers have used the concept of ACO to build upon a load balancing solution set within nodes of a cloud system. ACO can also reduce the delay in accessing the datasets in one cloud & apriori algorithm eliminates duplication of data sets.

2. BACKGROUND AND RELATED WORK

Earlier, the cost for data set storage can be minimised by using enhanced CTT-SP based algorithm and a novel runtime local-optimization based storage strategy. This is achieved through partitioning Data Dependency Graph (DDG) into Linear DDG segments. DDG is a directed acyclic graph (DAG), which is based on data provenance in scientific applications.In other words, it depicts the generation relationships of data sets,by which the deleted data sets can be regenerated from their nearest existing preceding data sets.This DDG is partitioned into linear segments and enhanced CTT-SP algorithm is applied. This approach is well worked for achieving minimum cost benchmark.

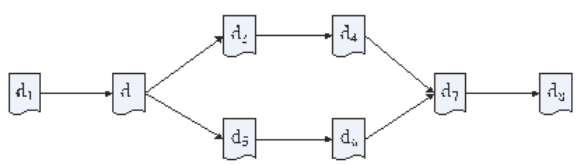

Fig. 1. A simple DDG.

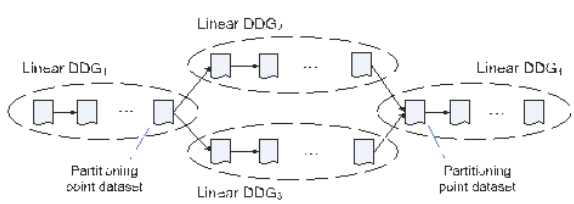

Fig. 2. Dividing a DDG into linear DDG segments.

However, there is not a proper algorithm used for minimising the data transfer cost from one cloud to another cloud.Based on this problem, we need to achieve two solutions: 1) Store the generated application data sets with a cost close to or even the same as the minimum cost benchmark, and 2) Incorporate the data transfer cost into minimum cost benchmark.

2.1 OVERVIEW OF ALGORITHMS

2.1.1 Usage of ACO in Cloud Computing:

The ACO is a probabilistic technique for solving computational problems which can be reduced to finding good paths through graphs. Earlier, ACO has been previously used by researchers in cloud computing for addressing load balancing, job scheduling and related problems. ACO is inspired from the ant colonies that work together in foraging behavior. In fact the real ants have inspired many researchers for their work and the ants approach has been used by many researchers for problem solving in various areas.

This phenomenon of the ants was used in many algorithms for optimization where the ants follow each other through a network of pheromone paths. The ants upon traversal from one node to another update the pheromone trail of that path, so a

Volume : 3 | Issue : 2 | February 2014 • ISSN No 2277 - 8179

path becomes more feasible if more ants traverse upon it. Paths that have the highest pheromone intensity have the shortest distance between the point and the best food source. The movements of these ants independently update a solution set.

In context of Cloud Computing the 'food' implies different web services. Ants will choose the path which is rich in pheromone. The reason is that the shorter path will receive a greater amount of pheromone than longer path because the ants choosing the shorter path will reach to food early and return to the nest by same path due to un-evaporated of pheromone.Therfore, the shorter path will have more amount of pheromone which finally results all ants to choose the shortest path..

Ants in the system can be traverse in two types:
1) **Forward movements**--It is the movement representing the forward direction of ants for extracting or searching the food sources.

2) **Backward movements**-It is the movement representing the direction of ants after picking up food from the food sources traverse back to the nest for storing their food.

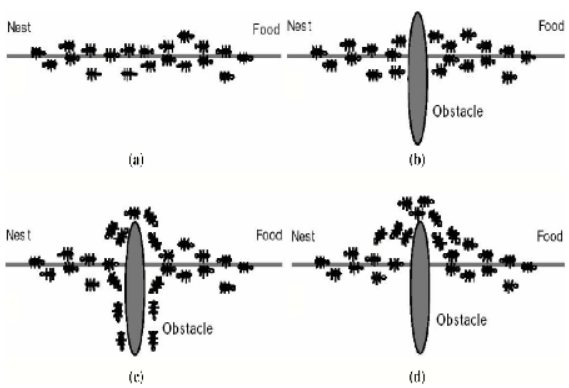

Fig.3. Modification in ants path upon encountering an obstacle

The concepts of ACO were applied in a more practical way where the pheromone values were updated by adding a prize, punishment and load balancing coefficient to the original values. The value of pheromones were updated after job completion in such a way that if job was completed successfully, the pheromone values were updated by the prize coefficients whereas if vice-versa they were updated by the punishment coefficient. Ant algorithms are one of the most popular examples of swarm intelligence systems, in which a number of ant inspired agents which are specialised in particular sophisticated functionality follow simple rules with no centralized control. The complex global behavior emerge from their local interactions using pheromone. Bio-inspired methods have already been started to solve a number of complex problems, such as routing, task allocation, graph partitioning, etc.

2.1.2 Usage of Appriori Algorithm in Cloud Computing:
The Apriori Algorithm is based on the Apriori property:
- If an item set I does not satisfy the minimum support threshold (i.e., P (I) < min_sup), then I is not frequent.
- If an item A is added to the item set I, then the resulting item set (i.e., I v A) cannot occur more frequently than I, i.e., P (I vA) < min_sup.

The Apriori algorithm is based on the support-confidence framework and the assumption that subsets of low-frequency itemsets must be low-frequency, use (k-1)-dimensional itemsets to generate the k-dimensional itemsets candidate. Though the candidate-base method prunes a lot of infrequent itemsets before calculates the occurrence rate, this algorithm is still time-consuming. Mining association rule consists of the following two steps: (1) finding the item sets that are frequent in the data set: The frequent item sets are set of those items whose support(*sup(item)*)in the data set is greater than the minimum required support(*min_sup*). (2)generating interesting rules from the frequent item sets on the basis of confidence (*conf*). There are several researches focused on the improvement of Apriori algorithm. Apriori-Growth Algorithm combines Apriori algorithm and FP-Growth to mine association rules.

3 PROPOSED WORK
The main aim of this paper is to reduce the data transfer cost and to incorporate into a minimum cost benchmark.For this, it should achieve a minimum shortest path to travel and reach the destination.Through ACO algorithm, ant travels through a minimum distance path, thereby reduces the cost of data transfer. The main task of ants in the algorithm is to redistribute work among the nodes. The ants traverse the cloud network, selecting nodes for their next step from the classical formula given below, where the probability *Pk* of an ant,which is currently on node r selecting the neighboring nodes for traversal, is:

$$Pk(r,s)=[\tau(r,s)][\eta(r,s)]\,\beta$$

$$[\tau(r,s)][\eta(r,s)]\,\beta$$

r = Current node,

s = Next node,

τ = Amount of pheromones in the edges,

η = The desirability of the ant movement (if the move is from an under loaded node to overloaded node or vice-versa the move will be highly desirable),

β = Depends upon the the movement distance with the relevance of the pheromone concentration.

However, higher the number of ants more frequent would be the data changes and load balancing and thus network efficiency. For this reason, we have to limit the number of ants in the network in order to keep the collection of fresh data and reduce variance, as well as to avoid congestion of the ants.

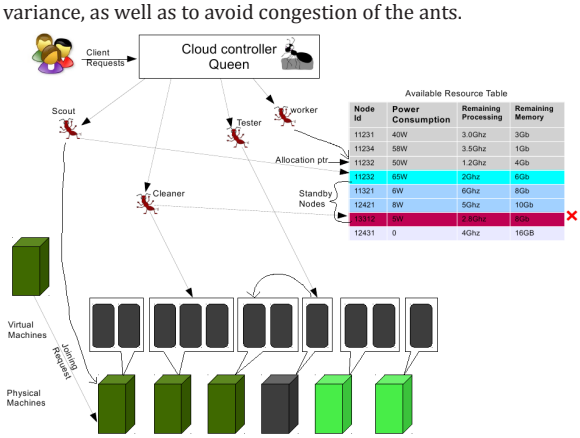

Fig.4. Cloud Architecture with Ant Agents

Users/Brokers: They submit service requests to the cloud via cloud controller for processing.

Cloud Controller: It work as the interface between the cloud service provider and external users/brokers.This is similar to the Queen in the ant colony.

Virtual Machines(VMs): This is where the applications of customers will be deployed. We can dynamically create, start, stop and migrate these VMs depending on our requirement, from one physical machine to another.

Physical Machines: These will provide hardware infrastructure for creating virtual machines.These are actually physical computing servers

In this approach, incoming ants update the entries in the pheromone table of a node. For instance, an ant traveling from source to destination will update the corresponding entry in the pheromone table in the node. Consequently, the updated routing information in it can only influences the routing ants and calls which has its destination. However, for asymmetric networks, the costs may be different. Hence, in this approach updating pheromone is the only appropriate method for routing in symmetric networks. If an ant is at a choice point when there is no pheromone, it makes a random decision. However, when only pheromone from its own colony is present there is a higher probability that it will choose the path with the higher concentration of its own pheromone type.

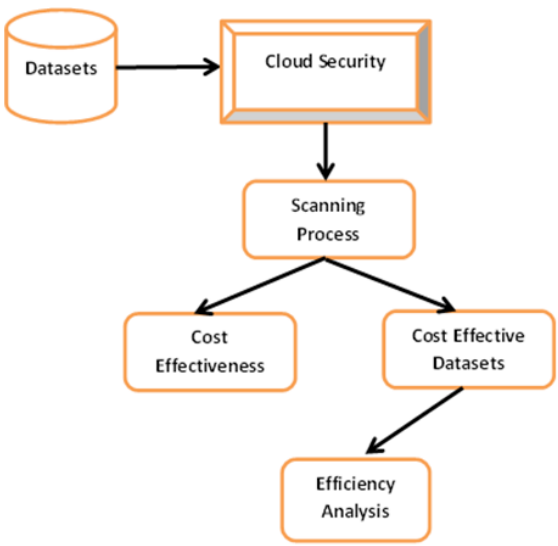

Fig.5. Overall Architecture

Datasets Storage : Association instructions display attributes that occur recurrently are collected in a given dataset.

Applied Cost Effective Datasets Storage Strategy: Substantial calculation power and storage capacity of cloud computing systems allow experts to organize computation and data concentrated requests without organization asset where large application datasets can be stored in the cloud. However, due to the datasets should be intentionally stored in order to reduce the overall application cost.

Cost Effectiveness : The data attributions in technical applications deleted datasets can be restored and as such we develop an original cost actual datasets storage approach that can repeatedly store correct datasets in the cloud. This approach reaches a contained finest transaction between computation and storage, in the meantime also attractive users tolerance of data accessing delay into reflection.

Efficiency Analysis : The additional compound cost representations and the perfect of approximating datasets usage occurrences need to be inspected with which our approach can be improved adjusted to different technical requests in the cloud. Additionally, the calculation complexity of our strategy needs additional exploration, where the cost of consecutively the strategy itself should also be occupied into deliberation.

3.1 Cloud Controller and Queen Ant

The customers' request consisting of the following, are given to the controller.
(i) Throughput(THPUT) (In %)
(ii) Avg. Response Time(RTIME)
(iii) Application Code
(iv) Operating System
Cloud controller maintains a queue(Q) for storing the service requests for hosting the applications. It enqueues each of the service request received, in this queue. It generates the tester, scout, cleaner and worker ants periodically. The ant agents movement can be modelled in the following way.

Except Queen & Worker each and every ant maintains a Visited Node list which is initially empty. Each node in the cloud maintains a list of neighbouring node's information. Whenever a particular ant reaches a node, it updates the controller about the current utilization and randomly chooses an unvisited neighbouring node. When all the nodes are covered, it makes the Visited Node list empty and continues again in the same way.

3.2 Worker Ant

Whenever a service request received in the queue, one of the worker ants creates a VM with a specific CPU processing power and memory etc, if accepted. So, worker ants are always looking in the queue to check if there are some pending requests to be processed. If such a request is found, it dequeues the request.

3.3 SLA Monitor Ant

It calculates the Avg. response time and throughput of the hosted application by continuously monitoring it. It passes this information to the hypervisor on that host in the form of a variable(SLAM) which is calculated depending on the performance of the application.

3.4 Tester Ant

The main job of the tester ants is to get the utilisation and power consumption information from each of the node and to update the available node's list. It also takes the load balancing decisions.

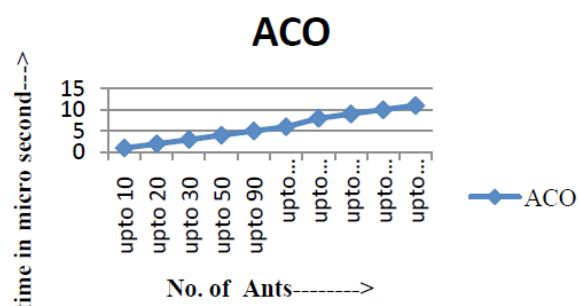

Fig 6. ACO in respect of Ant & Time

3.5 USES OF APRIORI IN CLOUD COMPUTING

In this section, we present how to redesign the Apriori algorithm so that it can be effectively executed in cloud computing. The Count Distribution (CD) algorithm is most direct parallelization of the Apriori algorithm. It uses the sequential Apriori algorithm in a parallel environment and assumes datasets are horizontally partitioned among different sites. The global candidate item sets and frequent item sets are stored in each node. In each step, the algorithm counts the support of candidate item sets of the local data using the apriori algorithm. After each scanning, it exchanges local support counts with all other nodes to generate a global support counts. The main advantage of Count Distribution algorithms is that it does not exchange data tuples between processors since it only exchanges the counts.

The output folder contains the frequent items with their count after the successful run of certain iterations. The outputs from the two successive iterations are read at a time. The confidence is calculated by using the count written along with the successive frequent item set. If the confidence is 100% the rule is considered to be an interesting one. For example, the frequent item set {*A,B,C*} from iteration 2 has *support_count*= 50 and *support_count* of {*B,C*} from iteration 1 is 50. The relation would be *{B,C} => {A}*, as the confidence is 100% and *support_count > min_sup*. The items in the frequent item set are comma separated. A frequent item set is read and its support count saved. The subsets of the frequent item set are found. The support count of the subsets is read from the previous iterations output files. Then the confidence is calculated. If the confidence is found to be 100%, the rule is generated as an interesting rule. The association rule

for later iteration frequent item set is given more importance than the previous ones.

4 CONCLUSION

In this paper we have proposed a method for achieving minimum data set storage cost along with minimum data transfer cost by modifying ACO with appriori algorithm.We will consider to develop this improved algorithm which can scale up to large data set with comparatively less cost. Moreover, this distributed algorithm can also cater to the distributed nature of the input data. We believe that this mechanism is very flexible and reliable and can be extended with improvements, since the solution modules are built as independent intelligent agents. We can incorporate additional functionalities in any of these ant agents. Furthermore, details of the management of the distributed systems, such as data transferring among nodes and node failures are taken care by Hadoop, which adds a great deal of robustness and scalability to the system.We believe that all these facilities will leads to minimization of data transfer cost in the cloud nodes.

REFERENCE

[1] D. Yuan, Y. Yang, X. Liu, G. Zhang, and J. Chen, "A Data Dependency Based Strategy for Intermediate Data Storage in Scientific Cloud Workflow Systems," Concurrency and Computation:Practice and Experience, vol. 24, pp. 956-976, 2012. | [2] Z I. Adams, D.D.E. Long, E.L. Miller, S. Pasupathy, and M.W. Storer, "Maximizing Efficiency by Trading Storage for Computation,"Proc. Workshop Hot Topics in Cloud Computing, 2009. | [3] J. Csorba, "Ant system for service deployement in private and public clouds," Proceeding of the 2nd workshop on Bio-inspired algorithms for distributed systems(BADS), Jun. 2010. | [4] D. Yuan, Y. Yang, X. Liu, and J. Chen, "On-Demand Minimum Cost Benchmarking for Intermediate Data Sets Storage in Scientific Cloud Workflow Systems," J. Parallel and Distributed Computing, vol. 71, pp. 316-332, 2011. | [5] B. Soumya, M. Indrajit, and P. Mahanti, "Cloud computing initiative using modified ant colony framework," in In the World Academy of Science, Engineering and Technology 56, 2009. | [6] B. Rajkumar, B. Anton, and A. Jemal, "Energy efficient management of data center resources for computing: Vision, architectural elements and open challenges," in International Conference on Parallel and Distributed Processing Techniques and Applications, Jul. 2010. | [7] D. Yuan, Y. Yang, X. Liu, and J. Chen, "A Local-Optimisation Based Strategy for Cost-Effective Data Sets Storage of Scientific Applications in the Cloud," Proc. IEEE Fourth Int'l Conf. Cloud Computing, pp. 179-186, 2011. | [8] S. Agarwala, D. Jadav, and L.A. Bathen, "iCostale: Adaptive Cost Optimization for Storage Clouds," Proc. IEEE Int'l Conf. Cloud Computing, pp. 436-443, 2011. | [9] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A View of Cloud Computing," Comm. ACM, vol. 53, pp. 50-58, 2010. | [10] R. Buyya, C.S. Yeo, S. Venugopal, J. Broberg, and I. Brandic,"Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the fifth Utility," Future Generation Computer Systems, vol. 25, pp. 599-616, 2009. | [11] D. Kondo, B. Javadi, P. Malecot, F. Cappello, and D.P. Anderson, "Cost-Benefit Analysis of Cloud Computing versus Desktop Grids," Proc. 23th Int'l Parallel and Distributed Processing Symp., 2009. | [12] X. Liu, D. Yuan, G. Zhang, W. Li, D. Cao, Q. He, J. Chen, and Y. Yang, The Design of Cloud Workflow Systems. Springer, 2012. | [13] D. Warneke and O. Kao, "Exploiting Dynamic Resource Allocation for Efficient Parallel Data Processing in the Cloud," IEEE Trans. Parallel and Distributed Systems, vol. 22, no. 6, pp. 985-997, June 2011. | [14] L. Young Choon and A.Y. Zomaya, "Energy Conscious Scheduling for Distributed Computing Systems under Different Operating Conditions," IEEE Trans. Parallel and Distributed Systems, vol. 22, no. 8, pp. 1374-1381, Aug. 2011. | [15] M. El-Hajj and O. Zaiane. Parallel leap: large-scale maximal pattern mining in a distributed environment. In Parallel and Distributed Systems, 2006. ICPADS 2006. 12th International Conference on, volume 1, page 8 pp., 0-0 2006. | [16] Nada M.A. AL-Salami, "System Evolving using Ant Colony Optimization Algorithm ", Journal of Computer Science 5 (5): 380-387, 2009, ISSN 1549-3636. | [17] X. Zhong and C.-Z. Xu, "Energy-Aware Modeling and Scheduling for Dynamic Voltage Scaling with Statistical Realtime Guarantee," IEEE Trans. Computers, vol. 56, no. 3, pp. 358-372, Mar. 2007. | [18] A. Li et al., "Cloudcmp: shopping for a cloud made easy," in Proc of the 2nd USENIX conference on Hot topics in cloud computing, 2010 | [19] H. Topcuouglu, S. Hariri, and M.-Y. Wu, "Performance-Effective and Low-Complexity Task Scheduling for Heterogeneous Computing," IEEE Trans. Parallel and Distributed Systems, vol. 13, no. 3, pp. 260-274, Mar. 2002. | [20] S.C. Wang, K.Q. Yan, W.P. Liao and S.S. Wang, Towards a Load Balancing in a Three-level Cloud Computing Network, Proceedings of the 3rd IEEE International Conference on Computer Science and Information Technology, pp. 108-113,2010. | [21] Z. Zhang and X. Zhang, A Load Balancing Mechanism Based on Ant Colony and Complex Network Theory in Open Cloud Computing Federation, Proceedings of the 2nd International Conference on Industrial Mechatronics and Automation, pp. 240-243, 2010. | [22] C. Gong, J. Liu, Q. Zhang, H. Chen and Z. Gong, The Characteristics of Cloud Computing, Proceedings of the 39th International Conference on Parallel Processing Workshops, pp. 275-279, 2010. | [23] C. Wang, K. Ren, W. Lou, and J. Li, "Towards Publicly Auditable Secure Cloud Data Storage Services," IEEE Network Magazine, vol. 24, no. 4, pp. 19-24, July/Aug. 2010. | [24] C. Wang, Q. Wang, K. Ren, and W. Lou, "Towards Secure and Dependable Storage Services in Cloud Computing," IEEE Trans. Service Computing, vol. 5, no. 2, 220-232, Apr.-June 2012. | [25] C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia, "Dynamic Provable Data Possession," Proc. ACM Conf. Computer and Comm. Security (CCS '09), pp. 213-222, 2009. | [26] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," NIST Special Publication 800-145, 2011. | [27] S. Di, C.-L. Wang, W. Zhang, and L. Cheng, "Probabilistic Best-Fit Multi-Dimensional Range Query in Self-Organizing Cloud," Proc. IEEE 40th Int'l Conf. Parallel Processing, pp. 763-772, 2011. | [28] X. Meng et al., "Efficient Resource Provisioning in Compute Clouds via vm Multiplexing," Proc. IEEE Seventh Int'l Conf. Autonomic Computing (ICAC '10), pp. 11-20, 2010. |