International Journal of Pure and Applied Mathematics
Volume 118 No. 20 2018, 311-319
ISSN: 1314-3395 (on-line version)
url: http://www.ijpam.eu
Special Issue

ijpam.eu

CONSISTENCY TUNING FOR QUORUM-REPLICATION DATASTORES FOR A VERSATILE SERVICE LEGAL AGREEMENT

M.Sindhu[1], Mr.S.Kannan[2], Dr.A.Muthukumaravel [3]
[1] Mphil. CS-Research Scholar, Department of MCA, BIHER, Chennai, Tamil Nadu, India
[2] Assistant professor, Department of MCA, BIHER, Chennai, Tamil Nadu, India
[3] Dean-Faculty of Arts & Science, & HOD-Department of MCA, BIHER, Chennai, Tamil Nadu, India

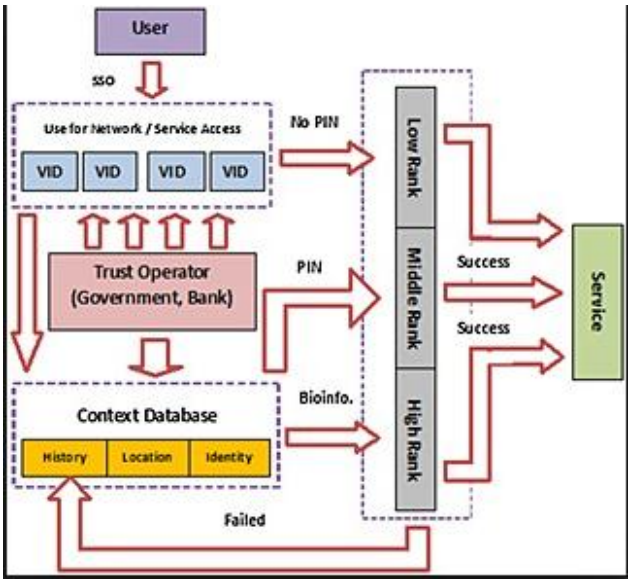
ABSTRACT:

For every storage operation an appropriate client-centric consistency setting are burdened by the users of distributed datastores that use quorum-based technique replication. It is finding very difficult for the above matching choice made because as it requires deliberating the trade-off within the latency and staleness, i.e., however stale (old) the result's. The application for a given operation is duly rely upon the client-centric consistency setting for the latency and staleness. We tend to present OptCon, a machine learning-based prognosticative framework that may modify the selection of client-centric consistency setting below user-specified latency and staleness thresholds given in the service level agreement (SLA). Below a given Service Legal Agreement, OptCon is what a java based wrapper which depicts a client-centric consistency setting that's mapping. Whereas manually tuned consistency settings stay fixed until explicitly reconfigured. OptCon tunes consistency settings on a per-operation basis with reference to dynamical employment and network state. we tend to demonstrate through an experiment that OptCon is a minimum of as effective as any physically chosen consistency settings in OptCon adapts to variations in employment, whereas a given manually chosen mounted consistency setting adapting to the SLA thresholds for various use cases. We tend

to additionally demonstrate that satisfies the SLA just for a characteristic employment.

INTRODUCTION:

Many quorum-based distributed data stores allow the developers to explicitly declare the desired client- centric consistency setting (i.e., consistency observed from the viewpoint of the client application) for an operation. Such systems accept the client-centric consistency settings for an operation in the form of a runtime argument, typically referred to as the consistency level. The performance of the system, with respect to a given operation, is affected by the choice of the consistency level applied. From the viewpoint of the user, the most important performance parameters affected by the consistency level applied are the latency and client-observed consistency anomalies, i.e., anomalies in the result of an operation observed from the client application, such as stale reads. Consistency anomalies are measured in terms of the client-centric staleness how stale (old) is the version of the data item (observed from the client application) with respect to the most recent version. According to the consistency level applied, the system waits for coordination among a specific number of replicas containing copies (i.e., versions) of the data item accessed by the given operation If the system waits for coordination among a smaller number of replicas, the chance of getting a stale result (i.e., an older version) increases. Also, the latency for the given operation depends on the waiting time for the above coordination; hence, in turn, depends on the consistency level applied. Hence, while choosing the consistency level, developers must consider how this choice affects the latency and staleness for a given operation.



LITERATURE SURVEY:

Distributed process groups on dynamostyle distributed storage

Deploying deduplication for distributed primary storage is a sophisticated and challenging task, considering that the demands of low read/write latency, stable read/write performance, and efficient space saving are all of paramount

importance. Unfortunately, existing schemes cannot present a satisfactory solution for the aforementioned requirements simultaneously. In this article, we propose D3, a dynamic dual-phase deduplication framework for distributed primary storage. Several major innovations are established in D3. First, we formulate a deduplication-oriented taxonomy called Dedup-Type, to group data with similar deduplication-related characteristics into larger categories. It serves as coarse-grained filter and one of the prioritizing references in D3. Second, D3 is a dual-phase framework-inline-phase and offline-phase deduplication processes work in concert with each other. Third, D3 operates in a dynamic manner. We design two critical mechanisms: context-aware threshold adjustment (CTA) for local inline-phase deduplication, and deferred priority-based enforcement (DPE) for global offline-phase deduplication. The CTA mechanism enables selective deduplication under a periodically updated threshold. Data skipped during the inline phase is regarded as a candidate for offline phase, and is handled in a prioritized order under the governance of DPE mechanism. Evaluation results demonstrate that, compared with conventional inline and offline deduplication schemes, D3 achieves more efficient and stabler read/write performance with competitive space saving.

Consistency-based service level agreements for cloud storage

Cloud computing uses Internet data centers to host applications and data storage. Cloud computing resources and services are offered to customers on pay-per-use model while the quality of the offered resources and services are defined using service level agreements also known as SLAs. Unfortunately, there is no standard mechanism to verify and assure that services delivered by the cloud provider satisfy the SLA agreement in an automatic way. To fill this gap we propose a framework for SLA assurance, which can be used by both cloud providers and cloud users. The proposed framework assesses performance of cloud applications with and without introducing system and component failures and then helps to resolve or mitigate failures to assure the required quality of cloud applications. The evaluation results obtained through simulations and using testbed experiments demonstrate good agreement with the design objectives.

Using cassandra for real-time analytics

The healthcare industry is changing at a dramatic rate. There are multiple processes going on within the health sector. These processes not only impact the care of individuals but also help medical practitioners and the delivery of care and services. The

industry can take advantage of big data analytics to ensure that all the multiple processes within the industry are running smoothly. Big data analytics is not just an opportunity but a necessity. Recently, big data stream computing has been studied in order to improve the quality of healthcare services and reduce costs by capability support prediction, thus making decisions in real-time. This paper proposes a generic architecture for big data healthcare analytic by using open sources, including Hadoop, Apache Storm, Kafka and NoSQL Cassandra. The combination of high throughput publish-subscribe messaging for streams, distributed real-time computing, and distributed storage system can effectively analyze a huge amount of healthcare data coming with a rapid rate.

Probabilistically bounded staleness for practical partial quorums

Operation availabilities of pessimistic quorum-based replication strategies are strictly upper-bounded. Probabilistically relaxing their strict consistency notion permits to overcome this bound by introducing probabilistic data replication strategies. These strategies allow the exploitation of the data consistency vs. operation availabilities trade-off. In our approach, we add a certain number of only probabilistically intersecting quorums to the originally strictly intersecting partially ordered quorum system in order to "closely conserve" their original characteristic behavior. As one expects, the number and nature of probabilistic quorums added have a strong impact on the particular trade-off. But furthermore, it turns out that the order in which probabilistic and strict quorums must be probed is also very important. We relate two general strategies of ordering the probabilistic and strict quorums of those probabilistic quorum systems and evaluate their impact on the data consistency vs. operation availabilities trade-off. The evaluation uses Markov chain-based steady state analysis applied to representative instances of probabilistic quorum systems with the adequate partial orderings.

EXISTING SYSTEM:

With the current progressive, the developers have to physically figure a mapping consistency level for a given operation at run time. Reasoning about the above choice is troublesome because of: 1) the large number of possible SLAs, 2) unpredictable factors like dynamic network state and varying workload that impact the latency and staleness, and 3) the absence of a well-formed mathematical relation connecting the above parameters. This makes automated consistency tuning under latency and

staleness thresholds in the SLA a highly desirable feature for quorum-based datastores.

DISADVANTAGE:

Experimental results demonstrate that OptCon is at least as effective as any manually chosen consistency level in adapting to the different latency and staleness thresholds in SLAs. Furthermore, we also demonstrate experimentally that OptCon surpasses any manually chosen fixed consistency level in adapting to a varying workload, i.e., OptCon satisfies the SLA thresholds for variations in the workload, whereas a manually chosen consistency level satisfies the SLA for only a subset of workload types.

PROPOSED SYSTEM:

We present OptCon, a framework that automatically determines a matching consistency level for a given operation under a given SLA. We introduce OptCon, a novel machine learning based framework, that can automatically predict a matching consistency level that satisfies the latency and staleness thresholds specified in a given SLA, i.e., the predicted consistency level is weak enough to satisfy the latency threshold, and strong enough to satisfy the staleness threshold. Decision tree and Random Forest are implemented using an open source machine learning suite.

ADVANTAGE:

The main advantage for using machine learning based framework has,

1. A fast multi-core based design space exploration and optimization.

2. It is very much useful to automate the selection and validation of the massive scale image data.

The main advantage for using decision tree learning has

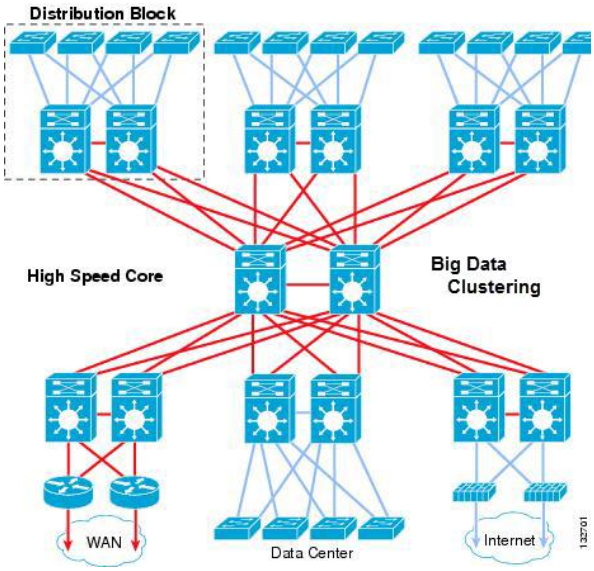
1. Clarity and transparency to the decision making process.
2. Ability to assign specific values to the problem.
3. It solves predictive analytics problem.

IMPLEMENTATION:

Distributed databases

The large number of possible use cases each comprising different SLA thresholds for latency and staleness, makes manual determination of a matching consistency level a complex process. The latency and staleness are also affected by the following independent variables (parameters): 1) the packet count parameter, i.e., the number of packets transmitted during a given operation, represents the network state the read proportion (i.e., proportion of reads in the workload), and 3) thread count. Parameters 2

and 3 represent the workload characteristics. We list the parameters in Table 1. Manually reasoning about all of these parameters combined together is difficult, even for a skilled and experienced developer.



Storage management

The effect of the consistency level on the observed staleness and latency, with respect to the independent variables. Based on their work upon the simplifying assumption that writes do not execute concurrently with other operations. Hence, it is not clear from to compute the latency and staleness from a real workload. Pileus and Tuba are the only systems that provide fine-grained consistency tuning using SLAs. But, instead of predicting, these systems perform actual trials on the system, and select the consistency level corresponding to the SLA row that produces maximum resultant utility, based on the trial outcomes. The trial-based technique can produce unreliable results due to the unpredictable parameters like network conditions and workload that affect the observed latency and staleness.

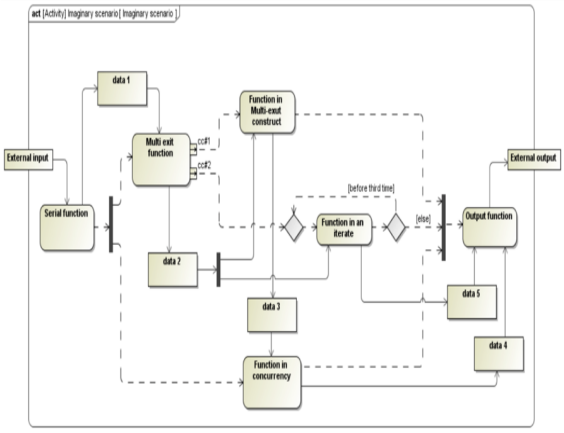
Distributed systems

In the absence of a closed-form mathematical model relating the consistency level to the observed latency and staleness, OptCon leverages machine learning-based prediction techniques following the footsteps of prior research. OptCon is the first work that trains on historic data, and learns a model M relating the consistency level and the input parameters (i.e., the workload and network state) to the observed latency and staleness. The model M can predict a matching consistency level with respect to the given SLA thresholds for latency and staleness, under the current workload and network state. A matching consistency level is weak enough to satisfy the latency threshold, and simultaneously strong enough to satisfy the staleness threshold.

Middleware

During the training phase, the Client module runs a simulation workload on the given quorum-based datastore. The Client calls the Logger module to collect the independent variables (parameters) and the observed

parameters for the operation from the JMX interface and appends these parameters into the training data. The Client then calls the Learner module, which trains the model from the training data applying machine learning techniques. During the prediction phase, the Client calls the Learner, with runtime arguments comprising the sub- SLA thresholds and the current values of the independent variables.



CONCLUSION:

OptCon automates client-centric consistency-latency tuning in quorum-replicated stores, based on staleness and latency thresholds in the SLAs. We implemented the OptCon Learning module using various Learning algorithms and do a comparative analysis of the accuracy and overhead for each case. ANN produces highest accuracy, and Logistic Regression the lowest overhead. Assigning equal weight to accuracy and overhead, the decision tree performed best. OptCon can adapt to variations in the workload, and is at least as effective as any manually chosen consistency setting in satisfying a given SLA. OptCon provides insights for applying machine learning to similar problems.

REFERENCES:

[1] A. Lakshman and P. Malik, "Cassandra: A decentralized structured storage system," SIGOPS Oper. Syst. Rev., vol. 44, no. 2, pp. 35–40, Apr. 2010. [Online]. Available: http://doi.acm.org/10.

1145/1773912.1773922

[2] C. Meiklejohn, "Riak PG: Distributed process groups on dynamostyle distributed storage," in Erlang '13.

[3] B. Calder, J. Wang, A. Ogus, N. Nilakantan, et al."Windows azure storage: A highly available cloud storage service with strong consistency," in Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles, ser. SOSP '11. New York, NY, USA: ACM, 2011, pp. 143–157. [Online]. Available: http://doi.acm.org/ 10.1145/2043556.2043571

[4] S.Ezhildevi, H.Shabuddeen, "Optimization Of Resource Provisioning Cost In Cloud Computing", International Journal of Innovations in Scientific and Engineering

Research (IJISER), Vol.1, no.3, pp.173-177, 2014.

[5] D. B. Terry, V. Prabhakaran, R. Kotla, M. Balakrishnan, M. K. Aguilera, and H. Abu-Libdeh, "Consistency-based service level agreements for cloud storage," in SOSP '13.

[6] D. B. Terry, M. M. Theimer, K. Petersen, A. J. Demers, M. J. Spreitzer, and C. H. Hauser, "Managing update conflicts in Bayou, a weakly connected replicated storage system," in SOSP '95.

[7] A. Jain, "Using cassandra for real-time analytics: Part 1," http:// blog.markedup.com/2013/03/cassandra-real-time-analytics/, 2011.

[8] P. Bailis, S. Venkataraman, M. J. Franklin, J. M. Hellerstein, and I. Stoica, "Probabilistically bounded staleness for practical partial quorums," Proc. VLDB Endow., vol. 5, no. 8, pp. 776–787, Apr. 2012. [Online]. Available: http://dl.acm.org/citation.cfm?id=2212351.2212359

[9] S. Sidhanta, W. Golab, S. Mukhopadhyay, and S. Basu, "OptCon: An Adaptable SLA-Aware Consistency Tuning Framework for Quorum-based Stores," in CCGrid '16.

[10] P. Flach, Machine Learning: The Art and Science of Algorithms That Make Sense of Data. New York, NY, USA: Cambridge University Press, 2012.

[11] W. M. Golab, M. R. Rahman, A. AuYoung, K. Keeton, J. J. Wylie, and I. Gupta, "Client-centric benchmarking of eventual consistency for cloud storage systems," in ICDCS, 2014, p. 28.

[12] P. Cassandra, "Apache cassandra use cases," http : // planetcassandra.org/apache-cassandra-use-cases/, 2015.

[13] C. Li, J. Leit˜ao, A. Clement, N. Preguic¸a, R. Rodrigues, and V. Vafeiadis, "Automating the choice of consistency levels in replicated systems," in USENIX ATC 14.

[14] M. S. Ardekani and D. B. Terry, "A self-configurable geo-replicated cloud storage system," in OSDI 14.

[15] K. C. Sivaramakrishnan, G. Kaki, and S. Jagannathan, "Declarative programming over eventually consistent data stores," in PLDI '15.

[16] L. Ravindranath, J. Padhye, R. Mahajan, and H. Balakrishnan, "Timecard: Controlling user-perceived delays in server-based mobile applications," in Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles, ser. SOSP '13. New York, NY, USA: ACM, 2013, pp. 85–100. [Online]. Available: http://doi.acm.org/10.1145/2517349.2522717