A Priority Queue Algorithm for the Replication Task in HBase

Changlun Zhang
Science School, Beijing University of Civil Engineering and Architecture, Beijing, China
Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun, China
Email: zclun@bucea.edu.cn

Kaixuan Wang and Haibing Mu
School of Electronics and Information Engineering, Beijing Jiaotong University, Beijing, China
Email: hbmu@bjtu.edu.cn

***Abstract*—The replication of the non-structure data from one data center to another is an urgent task in HBase. The paper studies the priority growth probability of the priority replication queue and proposed a dynamic priority replication task queue algorithm based on the earliest deadline first algorithm (EDF). The experiment results show that the proposed algorithm can balance the replication overhead between the high and low priority tasks and avoid the low priority task starving to death as well as ensure the high priority task's interests.**

***Index Terms*—non-structure data, HBsase, replication queue, earliest deadline first algorithm (EDF)**

I. Introduction

The interaction among users generates more and more non-structure data in Web 2.0 era. These non-structure data have no specific structure and cannot be described with some a certain format, such as the micro blogging message (including the @ and hyperlinks, pictures, etc.), the xml file and so on. Internet companies establish large amount of gigantic datacenters around the world to store these non-structure data. The number of hosts in a single data center can be several hundred to tens of thousands. Google has more than 50 data centers and 20 million servers [1] to store its customers' daily production of massive amounts of non-structured data around the world. It is a big challenge to manage and use these data, including data reading, data storing, data indexing, data addressing, interface of data configuration and management, particularly the data replication among multiple data centers is more urgent.

BigTable[2,3] is a distributed storage system developed at Google for managing structured data and has the capability to scale to a very large size: petabytes of data across thousands of commodity servers. BigTable has the ability to store structured data without first defining a schema provides developers with greater flexibility when building applications, and eliminates the need to re-factor an entire database as those applications evolve.

However, BigTable cannot manage structured data. HBase[4-6] is the Hadoop database, which is an Apache open source project whose goal is to provide Big Table like storage. HBase is for storing huge amounts of structured or semi-structured data. Data is logically organized into tables, rows and columns. Columns may have multiple versions for the same row key. The data model is similar to that of Big Table.

Similar to the traditional data transmission or routing services, different data replication tasks among distributed data centers are of different QoS requirements because column families may belong to different business, users, columns with different priorities and requirements for delay, bandwidth, and availability are various according to different business and data. Therefore, it is necessary to provide a replication task management mechanism based on task priority for HBase data replication protocols. It can implement dynamic priority management for cache queue of replication task.

Luo[10] presents a probability-priority hierarchical scheduling algorithm. Compared to the priority queue scheduling algorithm[11,12], the algorithm can on the one hand promise the time delay and data loss performance of high priority data packet groups and on the other hand improve the packet loss performance of low priority data packet groups.

In this paper, we propose a dynamic priority scheduling scheme based on the sequence of priority growth probability referring to the priority sequence of the earliest deadline first algorithm (EDF) [7-9]. The priority is divided into three levels: low, middle and high with correcting. The dynamic priority scheduling method can balance the replication overhead between the high and low priority tasks and avoid the low priority task starving to death as well as ensure the high priority task's interests

The rest of paper is organized as follows. Section 2 introduce the EDF algorithm which is the basis of our algorithm in the next section. The main work of the dynamic priority scheduling scheme is described in the Section 3. Section 4 gives the experiment an analysis of the proposed algorithm. Section 5 concludes the paper.

II. An Overview of EDF

The earliest deadline first scheduling algorithm(EDF) [7-9] is widely used as a priority scheduling algorithm. It calculates priority in accordance with the task deadline and assigned a higher priority to the task close to the deadline. The task with highest priority is promised to run at every moment. EDF achieves a dynamic priority scheduling algorithm for the deadline of the task in buffer queue may change as time goes by.

EDF algorithm can always obtain a feasible schedule as long as there is one. In other words, if EDF algorithm cannot generate a feasible schedule, there will be no other feasible schedule. In every new ready state, EDF selects the task with the earliest deadline from the task that is ready but not yet fully processed, and allocates the required resources to the task. The scheduler immediately recalculates the deadline of the task and gives new priority order as new tasks add. It deprives running tasks of control right on processors and decides whether to schedule a new task or not according to the new task's deadline. The new task may be processed immediately if its deadline is earlier than the current task. In accordance with the EDF algorithm, the processing of the interrupted task will resume later.

EDF algorithm has a simple necessary and sufficient condition to determine the schedulability: as long as the load of the periodic task set U is not greater than 1. EDF algorithm can generate feasible scheduling and has such characteristics as:

1) Task model: the same as RMS (Rate-Monotonic Scheduling);

2) Priority assignment method: Priority is dynamically allocated as the nearer to the deadline the higher it is;

3) Schedulability: If the task set meets $\sum_{i=1}^{n} \frac{C_i}{T_i} \leq 1$, the task is schedulable.

EDF scheduling algorithm has been shown to be the optimal dynamic scheduling with necessary and sufficient condition. It is of up to 100%CPU utilization with more online scheduling overhead than the RMS.

EDF scheduling algorithm is established based on the following assumptions:

1) The emptive cost is very small;

2) Only processing requirements are significant, the I / O, memory and other resource requirements can be ignored;

3) All tasks are independent, there is no priority relationship constraint among them.

These assumptions simplify the analysis of the EDF. Assumption 1 shows that the mission to seize at any time, this process is not preempted for any loss, can be restored at a later time, one task was to seize the number does not change the overall workload of the processor. Assumption 2 shows that there are no other factors that lead to complex problems except sufficient processing capacity to ensure performing tasks within the time limit to check the feasibility. Assumption 3 specifies that there does not exist a priority constraint relationship which means that the release time of the task is independent upon the end time of other tasks. As to system which is not met the above three assumptions, we need to take priority and exclusion constraints to solve the problem.

The EDF algorithm is the optimal dynamic scheduling algorithm with single-processor. The upper limit its schedulability is 100%, that is to say, if the EDF algorithm cannot schedule a task set reasonable on a single processor, then the other scheduling algorithm also cannot accomplish this task.

III. Probability Priority Growth Algorithm Based on EDF

A. Priority of Replication Task in HBase

In distributed database HBase, column family data being to replicate and synchronize may be of different importance because of requirements and business types. It is necessary to set different priority to different column family or its column according to the different QoS requirements [13-15] in order to distinguish data of different priority during data replication among data centers. An improved priority queue for HBase replication can be constructed according to the theory of EDF.

The priority of column family data may be broken down into its column. It may also be different because it comes from different user or time. The replicated data to be stored by different priority task may belong to one column family, so it may not only reduce the queue length but also make the send, store, and read more batches, continuous and rapid by merging the tasks storing the same column family.

In some implementation of the priority queue, the priority of each task is static configuration and won't change over time. When more high-priority tasks queued, low-priority task is likely to be "starved to death" for it is always unable to get service. According to the idea of EDF, the priority of task should be dynamically adjust and increase over time. The implementation of the dynamic priority acts as follows:

1) Each replication task joining the queue is set to an initial priority;

2) The priority of every task increases over every period;

3) Each time, the task of the highest priority in the queue may be selected to be replicated.

Assuming that there are a total of N priorities from 1 to N, 1 is the lowest priority and N represents the highest priority, the bigger the number the higher the priority.

According to the above conclusions, the priority of replication task in HBase column family should include the task's merging priority and the growing priority over time.

B. Priority of the Merger Task

The merging of multiple replication tasks is based on their same index of the storage location, that is, they belong to the same column family, but with different priorities. Obviously, the priority of the merged task is at least equal to the highest priority of original task to ensure that the original important task remains a high priority.

If the priority of task i is t_i and the maximum number of the task to merge is MAX, then, the priority T of the task 1,2, ... i, ... n after being merged is:

$$T=\max\{t_1, t_2,\ldots t_i,\ldots t_n\}\ (n\leq MAX) \quad (1)$$

C. Simple Priority Queue Algorithm Based on EDF

Assume that a total of N priority from 1 to N, where 1 is the lowest priority and N represents the highest priority. Priority of the task in the queue is increased by 1 every period in the simple priority queue based on EDF(SPQA). Each time the task of the highest priority is selected to execution. This dynamic priority avoids the task with low priority dying of starvation, but it is unfair for high-priority task that the priority of low-priority task is growing too fast.

For example, in the case of eight priorities from 1 to 8, the task of the initial priority setting as 5 may grow up to 7 after two periods in the buffer queue. Assuming that there comes a new task with priority of 7, it will not be replicated because the task with initial priority of 5 catches the chance. It seems unfair to the high priority task for the priority of the lower priority growing too fast task. The sensitivity to the time of the low priority is lowered to solve this problem which can reduce the growing rate of the low priority tasks as shown in Figure 1. We can construct a priority growth probability sequence $\{P_i,\}$ (i = 1,2, ..., N) based on the total number of priority N, which makes the low-priority tasks has a lower priority growth probability and high-priority task has a high priority growth probability.

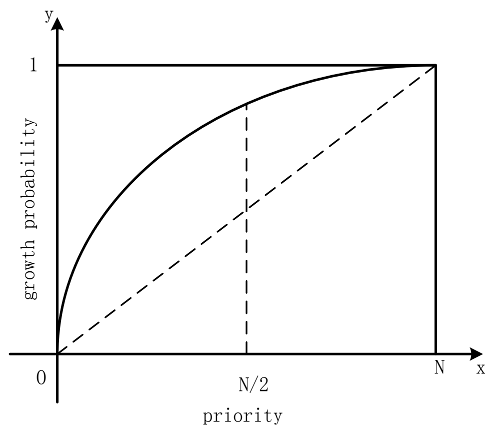

Figure 1. Priority growth probability

D. Probability Priority Growth Algorithm Based on EDF (PPGA)

In order to make the priority growth probability of the low-priority task lower and the high-priority task with high priority growth probability, we intends to construct a curve whose shape is similar to figure 2 to determine the priority growth probability. In the figure, the slope of the first half of the curve is greater than 1 while the latter half is less than 1. The curve is established with the following condition:

$$s=\sum_{\Delta x\to 0} y\Delta x=\int_0^N ydx\leq 1 \quad (2)$$

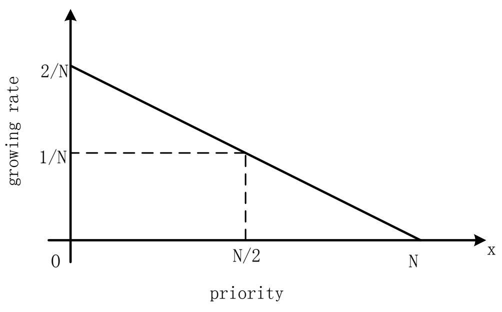

Figure 2. Reduce rate of of priority growth probability

Assuming that the linear function is y (x) = ax + b (0 ≤x ≤ N, x is an integer), we can obtain:

$$\begin{cases} a=-\dfrac{2}{N^2} \\ b=\dfrac{2}{N} \end{cases} \quad (3)$$

So,

$$Y(x)=-\frac{2}{N^2}x+\frac{2}{N}\ (0\leq x\leq N,\ x \text{ is an integer}) \quad (4)$$

Let $P_N = 1$, solve the priority growth probability of priority i as follows:

$$\begin{aligned} &P_N=1\\ &P_{N-1}=P_N-y(N-1) \quad (5)\\ &\ldots\ \ldots\\ &P_i=P_i+1-y(i) \end{aligned}$$

Add up the left and right part of the above equations respectively to get:

$$P_i=1-\sum_{k=i}^{N-1}y(k)=1+\frac{(N-i)(N+i-1)}{N^2}-\frac{2(N-i)}{N}(0\leq i\leq N) \quad (6)$$

For example, we construct a priority growth probability with eight priority and calculate the reduce rate of each priority growth probability by the equation (4) and get the priority growth probability of each priority by the equation (6). The results are shown in table 1 and figure 3.

Priority growth probability is not intuitive enough to obtain average period between the two priorities. It can be calculated as the expectations E (n):

$$\begin{aligned} E(n)&=1P_i+2(1-P_i)P_i+3(1-P_i)^2P_i+\ldots\\ &=P_i\sum_{k=1}^{\infty}k(1-P_i)^{k-1}\\ &=P_i(\sum_{k=1}^{\infty}k(1-P_i)^{k-1})'\\ &=\frac{1}{P_i} \end{aligned} \quad (7)$$

TABLE I.
PRIORITY GROWTH PROBABILITY

priority	Reduce rate	Growth probability	Average period
1	7/32	47/32	32/47
2	6/32	11/32	32/11
3	5/32	17/32	32/17
4	4/32	22/32	32/22
5	3/32	26/32	32/26
6	2/32	29/32	32/29
7	1/32	31/32	32/31
8	0	1	1

As shown in figure 3, the probability sequence constructed for the priority growth is incremental non-linear, priority growth probability of low-priority task is small while the high-priority task's is large to ensure its interests. It can be seen from the slope of eight broken line that low priority tasks has a small priority growth probability, but it has a high growing rate of growth probability, the slope slowly decreases with the priority growth. Low priority task can get a larger increase of the growth probability to ensure it will not be starved to death.

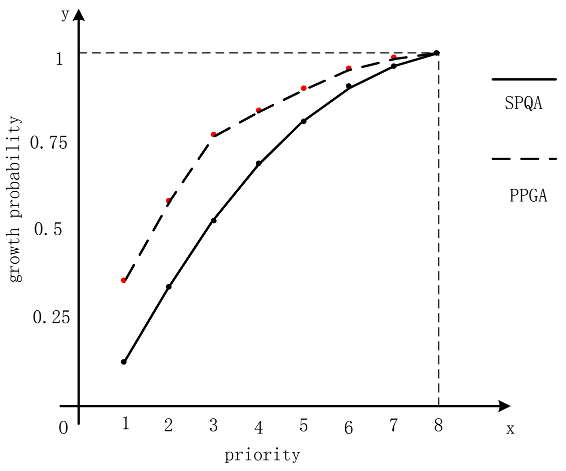

Figure 3. Priority Growth Probability of SPQA and PPGA

E. Analysis of Experimental Results

The effects of priority growth probability are compared in the following experiments. Here, the task which priority is 1, 4 and 7 are chosen from the heap of the maximum priority queue according to the priority growth probability.

As shown in figure 4, priority of the task in the queue is increased by 1 every period in SPQA and the priority of low-priority task is growing too fast. After eight periods, the task with high priority 7 increased to 8 by one period, the task with priority 4 increased to 8 by five periods, the task with lower priority 1 increased to 8 by eight periods.

However, the fast growing of priority is changed in PPGA. In figure 5, the task with high priority 7 still increased to 8 by one period; but the task with lower priority 1 has no change until the eighth period, and only increased to 2 lastly. PPGA balances the replication overhead between the high priority and low priority tasks.

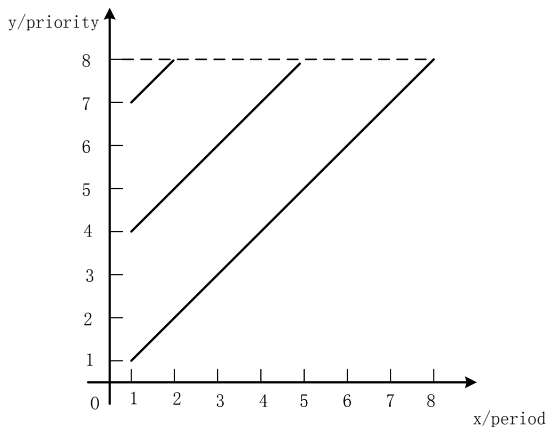

Figure 4. Priority Increase in SPQA

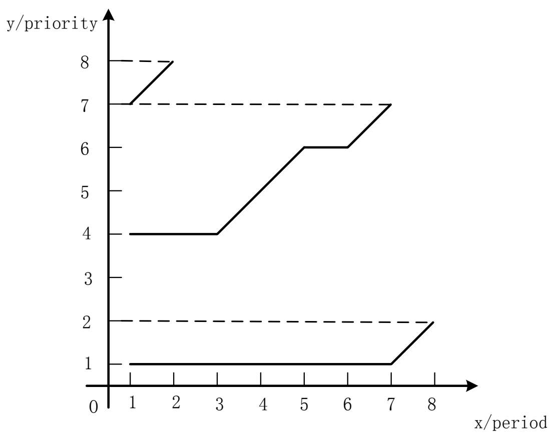

Figure 5. Priority Increase in PPGA

IV CONCLUSIONS

We studied the priority queue theory of the earliest deadline scheduling algorithm and the storing characteristics of column family in HBase in which the column family is the unit of replication tasks. A priority queue algorithm based on the priority growth probability is established, its priority growth probability sequence is evenly distributed in the interval [0, 1]. The probability growing rate of low priority is large and the high priority's is small. The algorithm balances the replication overhead between the high priority and low priority tasks and avoids the low priority task starving to death as well as ensures the high priority task's interests.

ACKNOWLEDGMENT

This work is supported by the work of the Beijing Municipal Organization Department of talents training-funded project (2010D005017000008) ,Beijing Institute of Architectural Engineering School research fund (Z10053) and Jilin University Key Laboratory of

Symbolic Computation and Knowledge Engineering of Ministry of Education research fund (93K-17-2012-02).

REFERENCES

[1] http://www.cnbeta.com/articles/73230.htm.
[2] Chang, F. and Dean, J. and Ghemawat, S, et.al. , Bigtable: A distributed storage system for structured data, ACM Transactions on Computer Systems (TOCS), 2008, 26(2): 4.
[3] Ankur Khetrapal, Vinay Ganesh, HBase and Hypertable for large scale distributed storage systems: A Performance evaluation for Open Source BigTable Implementations, from Internet.
[4] Dhruba Borthakur,The Hadoop Distributed File System: Architecture and Design, Available at http://wiki.apache.org/hadoop.
[5] HadoopDB Project, Available at http://db.cs.yale.edu/hadoopdb/hadoopdb.html.
[6] Azza Abouzeid, et.al., HadoopDB: An Architectural Hybrid of MapReduce and DBMS Technologies for Analytical Workloads, proceedings of VLDB '09, 2009, Lyon, France,pp922-933.
[7] Zhi Quan, Jong-Moon Chung, A Statistical Framework for EDF Scheduling, IEEE COMMUNICATIONS LETTERS, VOL. 7, NO. 10, OCTOBER 2003, pp. 493–495.
[8] Victor Firoiu, Jim Kurose,Don Towsley, Efficient Admission Control of Piecewise Linear Traffic Envelopes at EDF Schedulers, IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 6, NO. 5, OCTOBER 1998, pp. 558–570.
[9] Jianjun Li, et.al., Workload Efficient Deadline and Period Assignment for Maintaining Temporal Consistency under EDF, IEEE TRANSACTIONS ON COMPUTERS, pp.1-14.
[10] Luo huimei, Gao qiang, Song shuang, The study of hierarchical packer scheduling algorithm on probability-priority, Computer Applications and Software, 2011, 28(7), pp.57-59.
[11] Jiang Y, Tham C K,Ko C C, A probabilistic priority scheduling discipline for multi-service networks[C]. Proc of IEEE ISCC'01. Tunisia:[S n],2001, pp. 0450.
[12] Tham C K,Yao Q,Ko C C. Achieving differentiated services through multi-class probabilistic priority scheduling[J]. Computer Networks. 2002, 40(4):577-593.
[13] Guangjun Guo, Fei Yu, Zhigang Chen, Dong Xie. A Method for Semantic Web Service Selection Based on QoS Ontology. Journal of Computers, Vol 6, No 2 (2011), 377-386, Feb 2011.
[14] Elarbi Badidi, Larbi Esmahi.A Scalable Framework for Policy-based QoS Management in SOA Environments. Journal of Software, Vol 6, No 4 (2011), 544-553, Apr 2011.
[15] Bin Li, Yan Xu, Jun Wu, Junwu Zhu. A Petri-net and QoS Based Model for Automatic Web Service Composition. *Journal of Software*, Vol 7, No 1 (2012), 149-155, Jan 2012.

Zhang Changlun, was born in Jining Shangdong Province of China in 1972, earned Ph.D degree in Beijing Jiaotong University of China in 2009.
Now, He is a lecturer in Science School, Beijing University of Civil Engineering and Architecture. His research area focuses on networks information security , network public opinion and software engineer.