ISSN : 0974 - 7435 Volume 8 Issue 9

BioTechnology
An Indian Journal

FULL PAPER

BTAIJ, 8(9), 2013 [1246-1250]

# Study of application of replication methodology

**Cao Yonghui[1,2]**
[1]School of Management, Zhejiang University, (CHINA)
[2]School of Economics & Management, Henan Institute of Science and Technology, (CHINA)
E-mail : caoyonghui2000@126.com

## ABSTRACT

In this paper, We assume a primary-backup model with passive replication in combination with point-in-time replication yielding a one-way write to a read-only replica and develop a novel passive asynchronous replication methodology by applying the control approach. This new methodology is an optimization of the Primary-Backup replication service for creating a policy based backup. Finally, Solutions to the optimization problem used in the off-line processing are modeled as a zero-one Knapsack Problem (KP). The KP is a well-known and researched binary integer optimization model and was chosen for the simulation in this research.


## KEYWORDS

Worst-case data loss theorem;
Application replication model;
Application optimization model.

## INTRODUCTION

We seek a minimally intrusive method for increasing the resiliency of data in situations of limited or restricted processing and bandwidth resources. Given that all data is not valued the same in recovery from catastrophic events and that we have a dynamic system with limited or restricted processing and bandwidth resources where overloaded situations arise, is there a low impact method for increasing the resiliency of data? We assume a primary-backup model with passive replication in combination with point-in-time replication yielding a one-way write to a read-only replica. If we had clairvoyant knowledge of replication requests, we could schedule these requests to minimize the impact of a failure at any point in time based on a metric we define as the benefit. The increase in resiliency is based on the idea of minimizing the temporal distance between the primary and backup based on the value of the data given the constraints of the system. Such approaches have been proposed for static real-time primary-backup replication where worst-case bounds are known. In traditional systems or dynamic real-time systems reasonable bounds are not known, so we have two choices. Remove the uncertainty by queuing request until the end of an optimization interval, thus providing complete knowledge of the problem and solving using traditional means, or dealing with the uncertainty. Waiting to remove the uncertainty of future replication request increases the temporal distance between the primary and the backup, the parameter we are trying to decrease, so we must deal with the uncertainty.

This Section develops a novel passive asynchronous replication methodology by applying the control approach. This new methodology is an optimization of the Primary-Backup replication service for creating a policy based backup. The application of this approach to asynchronous replication uses an off-line multi-interval optimization to implement the user defined acceptance (control) policy for replications. The results of the optimization are used to train an MLP to enforce the acceptance policy in near real-time.

Solutions to the optimization problem used in the off-line processing are modeled as a zero- one Knapsack Problem (KP). The KP is a well-known and researched binary integer optimization model and was chosen for the simulation and Case Study in this research. The relative importance of the data is represented in the replication scheme through priorities. This ranking is based on the premise of the practice of user prioritization of the recovery process and the associated data. The priorities can be assigned a priori to the replication objects. The optimization maximizes the accrued benefit (priority). The objects identified as optimum for placement in the knapsack are accepted for replication. The trade-off in optimization of many lower priority smaller items being selected over a large higher priority item has the affect of creating a benefit ratio. The benefit ratio is defined as the priority value per unit of required bandwidth. The increases in resiliency are obtained by making decisions to replicate the objects with the greatest overall benefit to the system versus following the FCFS discipline. Other optimization models and schemes to represent data value may be used in the off-line processing as dictated by the specific application.

## APPLICATION FRAMEWORKS

A common approach to building fault recovery systems is the primary-backup model. In fault recovery systems characterized by bandwidth constraints or the acceptance of only minimal operational delays, employing a passive asynchronous replication solution following the primary-secondary replication scheme in the primary-backup model is common. In this scheme, the passive replication is an asynchronous refresh technology where the model distinguishes one replica as the primary server, which handles all client requests. A write operation on the primary server invokes the transmission of an update message to the backup servers, which updates the secondary replica(s). The backup replicas are read-only to avoid conflicts. In other words, there will be only one source for each replication object and this source controls the updates to the replica(s) in a one-way write to read-only framework. In order to decrease the impact of the replication on the operational system and ensure there are no temporal issues, often referred to as temporal variance, between data items, the updates are often determined by point-in-time dependant replication where each replication object will have a defined RPO which is considered acceptable from a data loss perspective.

A dependent replica tracks only data that has been over-written and are typically point-in-time images. Dependent replicas also known as sparse copies require additional storage that is roughly equivalent to the amount of data being written. Hitz provides further discussion of dependent replicas called snapshots in his discussion of Network Appliance’s Write Anywhere File Layout (WAFL) design. Figure 1 demonstrates the creation and update of a dependant replica.

The primary-secondary replication scheme replicates on a FCFS basis relative to the replication request. The FIFO queue implementing the FCFS policy requires little computing resources so the impact of the replication scheme on the operational system is almost negligible. When performing continuity of operations planning, the worst-case loss of data can be limited to two replication intervals for a replication object as long

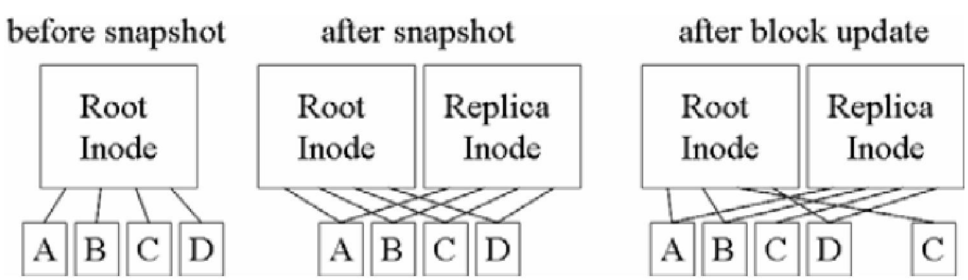


**Figure 1 : Dependent replica creation and update.**

as the system operates in an under- loaded or fully loaded condition, meeting the RPO for each replication object during all intervals.

Under the assumption that the system's best effort will be made by the system to met the RPO of a replication object, and given a fixed set of resources both within a system and required to complete a replication of an object during a period of time, the system will either be under- loaded, fully-loaded or overloaded during this period of time. The definition of fully-loaded is that the system has adequate resources to meet all requirements during an interval of time but no surplus. The definition of under-loaded is that the system has adequate resources to meet all requirements during an interval of time with surplus. The definition of overloaded is that there are inadequate resources to meet the current requirements of the system during an interval of time. In our case, these resources are assumed processing or bandwidth. In either case, if inadequate resources are present, the job cannot complete during the allocated time by definition, thus the RPO is missed for the given object. If adequate resources are present, either under- loaded or fully-loaded, the best effort assumption ensures the RPO will be met. Thus, the RPO of a replication object can only be met if and only if the system is under-loaded or fully-loaded.

## WORST-CASE DATA LOSS THEOREM

Theorem 1. *The worst-case data loss of a replication object upon failure of the primary data source is two replication intervals under the assumption that all RPOs are met.*

For each replication object requested for replication, there is an interval in which the replication is defined to occur. This interval is referred to as the replication interval and defined to be the time between replication requests of the same replication object. When a replication request is made, this defines a recover point for this instance of a replication request. The timing requirement of the request is considered met if the replication is completed before the expiration of the interval, thus a replication request successfully meeting its RPO requirement will at most be one replication interval out of synchronization with the primary. When a failure occurs, the failure may occur (1) during a replication or (2) during the processing time after the acknowledgement of the completion and before the next replication begins. In the base case, the first replication will complete or not. For case (1), the secondary is up to date and there is no additional data loss thus a worst-case data loss of one replication interval. For case (2), recovery must be from another source, as the replication has never succeeded. Under the assumption that all RPOs are met for n replication intervals, synchronizing the secondary to the primary at replication interval n-1, case (1) allows recovery during the n+1 replication interval to the just completed RPO for the $n^{th}$ replication interval, thus one replication interval of lost data. Case (2) requires the target of the replication to rollback to the last consistent state, i.e., the n-1 replication interval's RPO, and recover from that point, thus a worst case loss of two replication interval's data.

## APPLICATION REPLICATION MODEL

The data replication model used throughout this example consists of three main components: the replication requests, the adaptive decision-making algorithm and the actual replication. Processing elements make requests for replications. Each replication request is for the replication of a replication object. The replication protocol performs the replication between the primary and the replica controlled by the decision-making algorithm. The flow chart for the replication decisions used in this research is shown in Figure 2.

Replication objects are user defined containing one

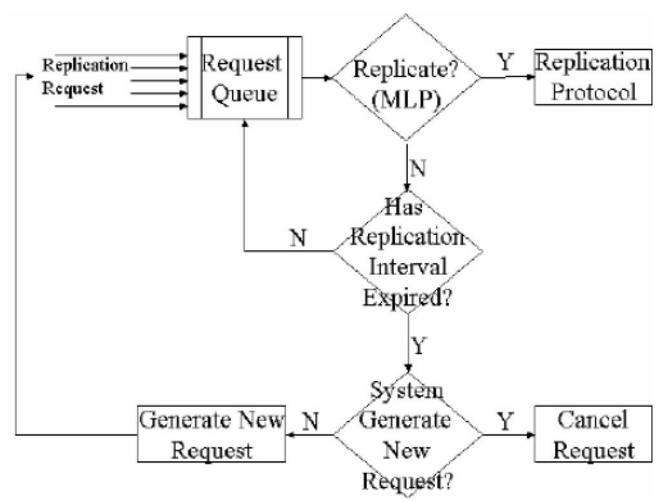


**Figure 2 : Flow chart of scheduling model.**

or more files and are assumed to contain all files within a service that have a temporal dependency. Each replication object is assigned benefit and a timing constraint. In this research, the benefit is defined to be a priority. The priority is a relative assessment of how important it is to have the data replicated during a given interval, the timing constraint. The definition of the benefit will be dependant on the objective function and optimization model used to implement the acceptance policy.

Another consideration of our replication model is how consistency is maintained. Since our target is read-only, the updates made to an object after a point in time (deltas) must be applied in order to maintain consistency. Two alternative solutions are evaluated.

For the first solution, deltas that are not accepted by the objective function are recalculated at each replication interval. This solution is dependant on the delta creation cost in the replication solution.

The second solution, in order for the scheduler to replicate a new delta, ensures that prerequisite updates have been completed. This is used in the data vault evaluation. An enhancement to the scheduler would be to make a single selection from a number of possible selections. The selections would contain one to all deltas. This can be modeled as a Multi-Choice Knapsack Problem (MCKP). Lee and Hsu [74] show how to use a neural network to solve the MCKP.

## APPLICATION OPTIMIZATION MODEL

For the simulations and Case Study developed in this Chapter, the solutions to the optimization problem used in the off-line processing are modeled as a zero-one KP. The KP is a well-known and researched binary integer optimization model. A description of the zero-one KP model follows. Keller, Pferschy and Pisinger [65] provide an overview and further discussion of the KP problem, its extensions and variations.

An instance of the zero-one KP can be defined by the capacity *C* and a set of *n* items where an item *i* is described by its profit $P_i$ and weight $W_i$. A subset of items is selected such that the total profit of the selected items is maximized and the total weight does not exceed *C*. The KP can be formulated as a solution for the following linear integer program:

Maximize $O = \sum_{i=1}^{n} P_i x_i$

Subject to $\sum_{i=1}^{n} W_i x_i \le C$

Where $o(n+\min\{2^{|C|}, C(|C|)\}) \cdots \{\pi_1, \cdots, \pi_m\}, i = 1, \cdots, n$

and $x_i \in \{0,1\}, i = 1, \cdots, n$

The profit $P_i$ belongs to the set of priorities $\{\pi_1, \cdots, \pi_m\}$. Higher priority values represent higher priority items. The decision vector $X = [X_1, \cdots, X_n]$ identifies which items are to be inserted into the knapsack. A value of one identifies insertion. Insertion into the knapsack indicates acceptance of the object for replication. All of the coefficients are positive integers and O is the objective function. The weight of each item is its bandwidth requirement for replication and must be less than the capacity so that it is possible to be scheduled. Capacity is the quantity of bandwidth available for replications during an interval. If an object's weight is greater than the current capacity, the object cannot be scheduled during the current interval. Finally, the sum of the weight of all items submitted to the scheduler must be greater than the capacity or all items are scheduled.

KP is NP-hard and can be solved a number of ways with one such approach being dynamic programming (DP). DP is a common approach as it provides solutions to KP in pseudo- polynomial time. The recursive formula for a DP solution to the zero-one KP is given in TABLE 1. The complexities in finding exact solutions to KP problems impact the scale to which the solutions are practical.

**TABLE 1 : Zero-one KP recursive formula.**

| | | |
|---|---|---|
| $F_{i,j} =$ | 0 | If $j \ge 0, i = 0$ |
| | $-\infty$ | If $j < 0, i = 0$ |
| | $F_{i-1,j}$ | If $j < w_i, 1 \le i \le n$ |
| | $\max\{F_{i-1,j}, (F_{i-1,j-w_i} + p_i)\}$ | If $j \ge w_i, 1 \le i \le n$ |

Pisinger [98] provides the minimal exact solution for solving the zero-one KP, which is used in this research. Pisinger's solution, the *MINKNAP* algorithm, is based on primal-dual dynamic programming algorithm, combined with a lazy kind of reduction and sorting. As detailed in the Pisinger paper, the *MINKNAP* algorithm is $o(n+\min\{2^{|C|}, C(|C|)\})$ where $|C|$ is the size of the minimal symmetrical core. For small core sizes |C| results in linear solution times, while difficult problems,

FULL PAPER

demanding a complete enumeration, are pseudopolynomially bounded.

## REFERENCES


**[1]** A.Bryson; Dynamic Optimization, Prentice-Hall, **(1998)**.

**[2]** D.Bertsekas; Dynamic Programming and Optimal Control, Athena Scientific, M.A.Belmont, **(1995)**.

**[3]** D.Bertsekas; Nonlinear Programming: 2nd Edition, Athena Scientific, M.A.Belmont,**(1999)**.

**[4]** D.Bertsekas, J.Tsitsiklis, Neuro-Dynamic programming, Athena Scientific, M.A.Belmont, **(1996)**.

**[5]** J.Lawrence, B.Pasternack; Applied Management science: A Computer-Integrated Approach for Decision Making, 2nd Edition, John Wiley& Sons, **(2002)**.

**[6]** P.Alsberg, J.Day; A principle for resilient sharing of distributed resources, In Proceedings of the Second International Conference of Software Engineering, SanFrancisco, California, October, 627-644 **(1976)**.

**[7]** D.Hitz, J.Las, M.Malcolm; File system design for an NFS file server appliance, In Proceedings of the USENIX Winter Conference, 235-246 **(1994)**.