Title

The reliability of regeneration-based replica control protocols

Authors

Long, DDE
Carroll, JL
Stewart, K

Publication Date

1989

DOI

10.1109/icdcs.1989.37978

The Reliability of Regeneration-Based Replica Control Protocols

Darrell D.E. Long*
Computer and Information Sciences
University of California
Santa Cruz

John L. Carroll
Kris Stewart
Department of Mathematical Sciences
San Diego State University

Abstract

The accessibility of vital information can be enhanced by replicating the data on several sites and employing a consistency control protocol to manage the copies. For many applications, the reliability of a system is a more important measure of its performance than its availability. These applications are characterized by the property that interruptions of service are intolerable and often involve interaction with real-time processes.

The reliability of a replicated data object depends on maintaining a viable set of current replicas. When storage is limited, it may not be feasible to simply replicate a data object at enough sites to achieve the desired level of reliability. If new replicas of a data object can be created faster than a system failure can be repaired, better reliability can be achieved by creating new replicas on additional sites in response to changes in the system configuration.

Several strategies for replica maintenance are considered, and the benefits of each are analyzed. Formulas describing the reliability of the replicated data object are presented, and closed-form solutions are given for the tractible cases. Numerical solutions, validated by simulation results, are used to analyze the trade-offs between reliability and storage cost. With estimates of the mean times to site failure and repair in a given system, the numerical techniques presented here can be applied to predict the fewest number of replicas required to provide the desired level of reliability.

1 Introduction

Distributed systems provide the opportunity to improve the fault tolerance of data objects through replication. The most common measures of fault tolerance include *reliability,* which is the probability that a replicated data object will remain continuously accessible over a given time period, and *availability,* which is the steady-state probability that the data object is accessible at any given moment. Availability has received much more attention, in part because its analysis is more tractable than that of reliability.

There are several reasons for favoring reliability as the primary performance measure. In many applications, the reliability of a system is a more important measure of its performance than its availability. These applications include process control, data gathering, and other tasks requiring interaction with real-time processes, where the data will be lost if not captured when it is available. The computers used for stock trading are a prime example: If these machines were to fail, the resulting chaos would halt trading.

The reliability of a replicated data object depends on maintaining a viable set of current replicas. Costs or space limitations may make it impossible to replicate a data object at enough sites to guarantee an acceptable level of fault tolerance. Since new replicas of a data object can generally be created faster than a system failure can be repaired, better reliability can be achieved by creating new replicas on other sites in response to site failures. This technique, known as *regeneration,* approximates the protection provided by additional replicas for a modest increase in storage cost.

The notion of regenerating replicas to replace those lost due to site failures was first proposed by Pu [19] as a technique for increasing the availability of replicated data objects in the *Eden* system [20]. His protocol, called the *Regeneration Algorithm,* provides mutual and serial consistency of replicated data objects in a partition-free distributed system. It creates new replicas of the data object to replace those lost due to system failure when it detects that one or more of the replicas have become inaccessible.

*Supported by faculty research funds from the University of California, Santa Cruz and by a University of California Regents Junior Faculty Fellowship.

CH2706-0/89/0000/0465$01.00 © 1989 IEEE

The Regeneration Algorithm is simple and efficient, but some weaknesses can be identified. It allows reads to continue as long as one current replica of the object remains accessible, while writes are disabled if fewer than the initial number of replicas are accessible and there are insufficient spares to replace the missing replicas. The protocol specification leaves the procedure for recovering from a total system failure unspecified, requiring manual intervention in the event of a total system failure. It is also unable to guarantee mutual consistency in the presence of network partitions. This approach is suitable for single segment carrier-sense networks, but fails on the increasingly more common multi-segment networks.

Noe and Andreassian have suggested that the adoption of an approach similar to the Available Copy [1,2] protocol would alleviate the problem of poor reliability for writes since it would allow writes to occur as long as a single replica of the data object remains available [17]. This suggests that using regeneration to maintain a viable set of current replicas is a technique that can be adapted to many existing replica control protocols [10]. By combining regeneration with protocols with desirable characteristics, the weaknesses of the Regeneration Algorithm can be successfully addressed.

In this paper, the application of regeneration to several replica control protocols is explored. Regeneration can be used with the Available Copy protocols [1,2,3,12] to improve fault tolerance in non-partitionable computer networks. When the communications network is susceptible to partitionings, consensus protocols are required. By combining regeneration with static majority consensus, a simple protocol for maintaining mutual consistency among replicas of a data object is obtained. Regeneration can be combined with the Dynamic Voting protocols [4,9,13] to provide an increased level of fault tolerance over that of static consensus protocols.

2 Applying Regeneration

A replica control protocol based on regeneration begins with a set of replicas placed on sites around the computer network. As these replicas fail, other sites (called spares) are located and new replicas are created on them by the regeneration protocol. Once a failed system component has been repaired, the storage used by the extra replicas can be relinquished.

For all of the protocols, it is essential that the regeneration protocol be atomic to ensure a consistent generation of replicas. In the case of quorum-based protocols, generations with fewer than a quorum could result if this condition is not met. In this event, the replicated data object would become permanently inaccessible since no quorum could ever be formed. It is assumed that an appropriate commit protocol [22] is used to ensure atomicity.

2.1 Regenerative Available Copy

An Available Copy protocol increases the reliability of the protocol by continuing to allow writes to occur as long as one replica of the data object remains accessible and by allowing reads from any available copy. By combining the Available Copy protocol with regeneration, better reliability than the Regeneration Algorithm is obtained since writes are allowed to continue as long as a replica of the data object remains available.

The Optimistic Available Copy protocol [3,12] provides a good basis for integrating regeneration with an Available Copy protocol. It provides all the facilities that are required to combine regeneration with an available copy protocol. In particular, its was-available sets, which are used to speed recovery from total failure by tracking the last site to fail, provide all the information that is needed to identify the set of replicas comprising the current generation. The was-available set for an available site represents those sites which received the most recent change to the replicated data object. This includes the set of all sites that received the most recent write and all of those sites which have repaired from that site.

In order for the Optimistic Available Copy protocol to allow a regeneration to occur, at least one replica must be in an available state and there must be at least one spare. The state of the replicated data object is first copied to the spare sites. The names of those spare sites are then included in the was-available sets of all available sites. The version number associated with the regenerated replicas is the current version number of the replicated data object.

The recovery procedure operates by using the information contained in the was-available sets stored at each site to determine the set of sites that failed last. This set of sites can be found by computing the closure of the was-available set of the recovering site. When a site recovers from a failure and finds a site that has been transformed into a spare in its was-available sets, it treats the spare as a failed site. The result is that the recovering site will be unable to compute the closure of its was-available set and so must wait for the last site to fail to recover.

When a recovering site establishes communication with an available site, it may find that a full complement of replicas are already available. In this case one

of the extant replicas can be destroyed and the storage reclaimed, thus reducing the amount of storage consumed. As with all regeneration-based protocols, the question of which replicas should be retained remains open. It would seem that the best choice would be to keep replicas on sites with the best reliability characteristics, but factors such as communication costs may make other choices more appropriate.

In order to show that Optimistic Available Copy with regeneration is correct, it is necessary and sufficient to prove that the replicated data object will not be made available following a total failure of the system until the last site to fail can be determined. The following theorem demonstrates this result. The proofs for all theorems in this paper can be found in Long's doctoral dissertation [10].

Theorem 2.1 *When using the Optimistic Available Copy protocol with regeneration, a site recovering from a total failure will not enter an available state until the last site to fail has been found.*

2.2 Regenerative Static Voting

Combining regeneration with static majority consensus requires the notion of generations in order to determine the current set of replicas. A generation is defined as the set of replicas that have participated in a particular regeneration. Each replica in the set is tagged with a *generation number*. By using generations, the current set of replicas can easily be determined and only that group is allowed to participate in quorum collection. The generation numbers used here are similar to those used by Pâris in his article on *voting with tokens* [18].

The protocol is a simple extension of static Majority Consensus Voting [6]. A majority is considered to be a majority of the votes assigned to the original set of replicas. As spares replace failed sites, they can be assigned the votes of the failed sites, ensuring that the net number of votes in any generation is never more than the original number of votes. Of course, it is possible to reassign votes using an appropriate protocol, but that is orthogonal to this discussion.

When the protocol determines that there are fewer than the desired number of replicas of the data object, a regeneration will be initiated. A regeneration cannot occur unless a quorum of the replicas can be collected. These replicas must be members of the current generation, which is indicated by the current generation number. If a quorum exists and there are spare sites available, some are chosen to hold the new replica of the data object. The state of the replicated data object, including the data and the version and generation numbers, is copied to the spare sites. The spares are transformed into full replicas and all participating sites will increment their generation numbers in order to disenfranchise any sites that did not participate. This preserves mutual consistency by excluding the replicas that did not participate in the regeneration from taking part in any future quorum.

When a vote is called, excess replicas of the data object will have obsolete generation numbers. These replicas can be transformed into spares since they are not members of the current generation and so cannot participate in any quorum.

In order to demonstrate the correctness of Regenerative Majority Consensus, it is necessary and sufficient to show that at any time there will be at most one generation of replicas that can constitute a quorum. The following theorem is similar to a result demonstrated by Pâris [18].

Theorem 2.2 *Under static majority consensus with regeneration, the protocol for creating a new generation of replicas will disenfranchise a minority of the current generation. Furthermore, the only quorum present will be made up of sites with the current generation number.*

There is no need for a site recovery protocol, since a site that recovers from a failure can determine if it is a member of an earlier generation by examining its generation number when the next quorum collection occurs. If found to be a member of an earlier generation, it can be transformed into a spare.

2.3 Regenerative Dynamic Voting

The Optimistic Dynamic Voting protocol [13] is an implementation of Dynamic Voting [4], similar to Dynamic-linear Voting [9]. It is more amenable to regeneration due to its use of partition sets, which are used to determine the required quorum for the next access operation. A partition set is maintained at each site, and represents the set of sites that participated in the last successful operation that included that site. The partition sets will be maintained when either a read or write operation occurs, and are updated when a site recovers from a system failure.

The partition sets contain all the information that is needed to identify replicas and provide a mechanism for excluding sites that are members of out-of-date quorums. Accomplishing a regeneration requires a majority of the replicas in the quorum set to be present. The definition of a quorum remains unchanged. Spare sites are selected, transformed into replicas, and the names of these sites are entered into

the partition sets of the quorum. The operation number of the replicated object is then incremented. Since a quorum must be present, incrementing the operation number has the effect of disenfranchising those replicas that did not participate in the regeneration.

There is only a slight change in the site recovery protocol. When a site recovers from a failure and finds that the original number of replicas are already present, then the storage that it consumes is superfluous and it can be transformed into a spare.

In order to demonstrate the correctness of Regenerative Optimistic Dynamic Voting it is necessary and sufficient to show that at any time there will be at most one generation of replicas that can constitute a quorum. This occurs because the act modifying the quorum provides the mechanism for disenfranchising sites.

Theorem 2.3 *Under Optimistic Dynamic Voting with regeneration, there is at any time at most one majority partition.*

3 Reliability Analysis

While the availability of a protocol measures the performance of that protocol over a long period of time, its reliability estimates the probability a replicated data object managed by that protocol will remain continuously available over a given period of time. In general, the reliability $\mathcal{R}_P(n, m, t)$ of an n-site system with m spares and managed by protocol P is defined as the probability that the system will operate correctly over a time interval of duration t given that an initial complement of n replicas and m spares were operating correctly at time $t = 0$.

Since availability is a measure of the steady-state properties of a system, the availability of replicated data objects has been extensively studied. By contrast, reliability is a measure of the behavior of a system under transition, and its analysis is much less tractable.

Because Optimistic Dynamic Voting and Dynamic-linear Voting have the same performance, and since Optimistic Available Copy is an efficient implementation of Available Copy, Dynamic-linear Voting (DLV) and Available Copy (AC) will be used for the remainder of this study.

3.1 System Model

The replicas of the data object are assumed to reside on distinct sites of a computer network, and these sites have independent failure modes, but have identical failure, repair, and regeneration characteristics.

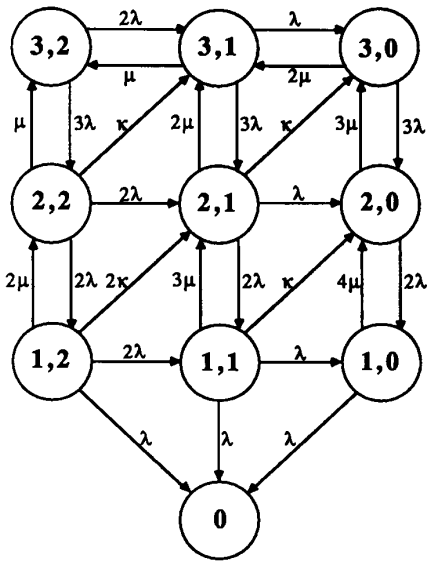

Figure 1: State Transition Diagram for Available Copy with 3 sites and 2 spares

The communication network connecting the sites is assumed to be reliable. The time to notice a site failure and complete a repair is assumed to be exponentially distributed with mean $1/\mu$. This includes the time to ascertain if this site is still intended to hold a replica of the data object and (if necessary) copy the data. Site failures are assumed to be exponentially distributed with mean rate λ. Site regeneration is similarly modeled by an exponential distribution with mean κ, which reflects the time to ascertain that a site has failed, verify that both the replicated data object and a suitable spare is available, and install a replica on that spare site.

The differential equations describing the behavior of systems managed by the replica control protocols can be derived from the state-transition flow rate diagrams. The states in the Markov chain are labeled to reflect the number of sites that can successfully respond to a request for the replicated data object. An n-site system with an unlimited number of spares is in state $\langle 0 \rangle$ if the replicated data object has been inaccessible at some point in the past, while for $1 \leq i \leq n$, the system is in state $\langle i \rangle$ if the object has been continuously accessible and if i replicas of the data object are currently accessible. Thus, no transitions are permitted from state $\langle 0 \rangle$, since only the behavior of the system prior to the first total failure is of interest. Flow rates to adjacent states are governed by the number of sites operational and the number under re-

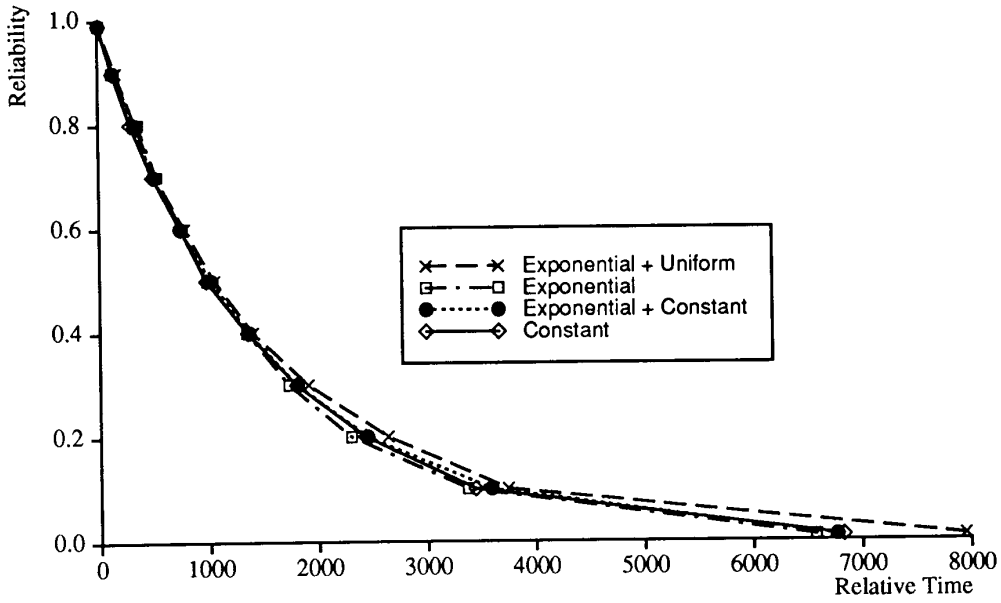

Figure 2: Compared Regeneration Distributions for Available Copy with 3 sites and 2 spares

pair.

A system maintaining n active sites with an additional m spare sites is in state $\langle j, k \rangle$ if j replicas are immediately accessible and k sites are currently available as spares. The state $\langle 0 \rangle$ will again denote the inaccessible state. The flow rate diagram for three sites with two spares managed by the Available Copy protocol is shown in Figure 1. Diagrams for Dynamic-linear Voting and Majority Consensus Voting are more complex but similar [11].

3.2 Analytic Results

For small numbers of sites, closed-form solutions for the reliability of some of the protocols can be obtained from the differential-difference equations. Less tractable systems can be both simulated and solved numerically. Simulation is crucial to characterizing site regeneration as a Poisson process: a site failure is generally discovered only after a non-trivial period of time. Since κ reflects the time necessary to both detect a site failure and restore the data base, exponential distributions are at best an approximation.

As shown in Figure 2, the reliability of these systems is relatively insensitive to the shape of this distribution. The data points shown were obtained by simulating the repairs and failures of a system of n sites until all sites failed, and noting the time at which the protocol would first deny access to a replicated data object. The process was repeated 1000 times, and the results were sorted to obtain an approximation of the reliability function. The resulting deciles reflect several regeneration distributions with identical first moments but disparate higher moments.

3.2.1 Unlimited Spares

It is only possible to derive closed-form solutions in the most elementary cases. This occurs when a protocol such as Regenerative Available Copy is assumed to have an unlimited supply of spares available.

The set of differential-difference equations arising from n sites managed by Regenerative Available Copy with an infinite number of spares, as shown in Figure 3, is given by

$$\begin{aligned}
\frac{dp_n}{dt} &= \kappa p_{n-1}(t) - n\lambda p_n(t), \\
\frac{dp_j}{dt} &= (j+1)\lambda p_{j+1}(t) + \\
&\quad (n+1-j)\kappa p_{j-1}(t) - \\
&\quad (j\lambda + (n-j)\kappa) p_j(t), 1 < j < n \\
\frac{dp_1}{dt} &= 2\lambda p_2(t) - (\lambda + (n-1)\kappa) p_1(t)
\end{aligned}$$

and

$$\frac{dp_0}{dt} = \lambda p_1(t)$$

with initial conditions

$$p_i(t) = \begin{cases} 0 & 0 \leq i < n \\ 1 & i = n \end{cases}$$

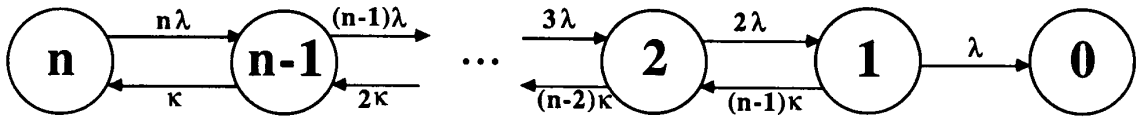

Figure 3: State Transition Diagram for Available Copy with n Sites and Unlimited Spares

In this system, $p_0(t)$ represents the probability that the system has failed by time t. The reliability of the system is $\mathcal{R}_{AC}(n,t) = 1 - p_0(t)$. When $n = 2$, the time-dependent solution to this Markov process yields

$$\mathcal{R}_{AC}(2,t) = \left(\frac{(3\lambda+\kappa)\sinh\left(\frac{t\sqrt{\lambda^2+6\kappa\lambda+\kappa^2}}{2}\right)}{\sqrt{\lambda^2+6\kappa\lambda+\kappa^2}} + \cosh\left(\frac{t\sqrt{\lambda^2+6\kappa\lambda+\kappa^2}}{2}\right)\right)e^{-\frac{(3\lambda+\kappa)t}{2}}$$

when failed sites are simply replaced from an unlimited supply of spares. In the presense of a total failure, no copies of the data object are available to regenerate a spare, and the replicated data object will be inaccessible until the critical sites are repaired. This does not affect the analysis of the reliability of the system, since it is only the behavior of the system prior to a total failure that is of interest. Similar systems of equations arise from the state-transition rate diagrams associated with Majority Consensus Voting and Dynamic-linear Voting.

3.2.2 Finite Spares

The state-transition rate diagrams for finite spares can lead to complex sets of equations. These systems of linear, constant coefficient ordinary differential equations (ODEs) are of the form $P'(t) = AP(t)$, with initial condition $P(0) = P_0$. The solution is given analytically by

$$P(t) = e^{tA}P_0$$

where e^{tA} denotes the matrix exponential [16].

For simplicity of exposition, assume A has full geometric multiplicity. Its Jordan canonical form $A = T\Lambda T^{-1}$ consists of the diagonal matrix Λ with eigenvalues, λ_i, of A on the diagonal and T, whose columns are the eigenvectors of A. The ODE can then be diagonalized:

$$P'(t) = T\Lambda T^{-1}P(t)$$

Defining $Z(t) = T^{-1}P(t)$, the differential equation is $Z'(t) = \Lambda Z(t)$, with solution $Z(t) = e^{t\Lambda}Z_0$, where $e^{t\Lambda}$ is the diagonal matrix with entries $e^{t\lambda_i}, i = 1, ..., n$. The general solution is thus

$$P(t) = Te^{t\Lambda}T^{-1}P_0$$

which can be evaluated at any point t in time.

This procedure is costly. The vector $Z_0 = T^{-1}P_0$ need only be computed once, requiring approximately $\frac{1}{3}n^3$ flops, where a flop is a floating point add coupled with a floating point multiply. The n exponentials $e^{t\lambda_i}$ that comprise $e^{t\Lambda}$ would be formed for each value of t of interest. The cost would be reduced by computing the solution at equally spaced points $t_k = k \cdot \Delta t$ using $e^{t_{k+1}\lambda_i} = e^{\Delta t\lambda_i} \cdot e^{t_k\lambda_i}$. The propagation matrix

$$e^{\Delta t\Lambda} = \begin{pmatrix} e^{\Delta t\lambda_1} & 0 & \dots & 0 & 0 \\ 0 & e^{\Delta t\lambda_2} & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & e^{\Delta t\lambda_{n-1}} & 0 \\ 0 & 0 & \dots & 0 & e^{\Delta t\lambda_n} \end{pmatrix}$$

need only be formed once, and later time step values can then be formed recursively from the previous step beginning with P_0 and using

$$P_{k+1} = T \cdot e^{\Delta t\Lambda} \cdot P_k$$

at a cost of one matrix-vector multiply (n^2 flops) per step. The major cost, though, arises when the eigensystem of A is computed.

Obtaining the eigensystem of A is equivalent to finding the roots of its characteristic polynomial. It was shown by Evariste Galois [8] that there is no direct method possible for computing the roots of a polynomial of degree higher than 4. This implies that models with 5 or more states will require an iterative process to obtain the eigensystem. The most effective is the QR algorithm [7]. Actual convergence of the iterative QR algorithm depends on the problem and the conditioning of the eigenvectors, but this one-time cost is estimated at $15n^3$ flops [7].

Thus, computing the matrix exponential directly is very costly and alternate methods of solution are desirable. Similar problems are typically attacked with general purpose ODE solvers such as RKF45 [5], based on the Runge-Kutta-Fehlberg method. This robust, reliable piece of mathematical software is capable of solving a general, non-linear system of initial value problems. It does not take advantage of the linear, constant coefficient nature of this particular problem, but is readily available and effective

for small models. There is a fundamental limitation for this method since there are no transitions leaving the failed state. The resulting system of linear ODEs must have a zero eigenvalue. Because the remaining eigenvalues are all negative, the system is "infinitely stiff" [21] and RKF45 is very inefficient. Either an approximation using the Padé expansion [7] of $e^{\Delta t A}$ or, since the eigenvalues are all real and nonpositive, a Taylor series for $e^{-\Delta t A}$ is more cost effective. Accuracy is easily controlled by adjusting the size of $\Delta t A$.

3.2.3 Results

Though the eigensystem technique is not as cost effective as Padé, the systems modeled in this paper are small and cost effectiveness is not paramount. The eigenvalues provide additional information on the decay constants for the various models. The numerical calculations were all produced using this more costly method to compute the exact solution of the differential equations.

To determine the relative reliability afforded by each protocol, a three site, two spare system was analyzed. Numerical solutions, validated with simulation results, were obtained for each of the protocols under identical conditions: $\lambda = 0.1, \mu = 1.0, \kappa = 10$. In the following figures, the discrete points reflect the deciles found by simulation, while the curves represent the exact numerical solutions.

Figure 4 illustrates the marked advantage of Available Copy over both quorum-based protocols, thus making it the protocol of choice in an environment in which network partitions are impossible. When partial communication failures can occur, Dynamic-linear Voting clearly surpasses Majority Consensus Voting. The relative ordering of the protocols agrees with the non-regenerative case [15], and further analysis shows that these relative advantages are independent of the number of sites and spares. These conclusions are also independent of the rate of regeneration κ [11].

To examine the trade-off between storage space and reliability, the performance of the Available Copy protocol was compared for each reasonable allocation of sites and spares in a five-site system. Figure 5 shows that replacing a single replica with a spare has only a slight detrimental effect on the reliability of the data object. Further decreasing the number of replicas markedly degrades the reliability. Similar limits are reached for the other protocols as well.

4 Conclusions

Replica control protocols that employ regeneration increase the reliability and availability of replicated data objects by creating new replicas when one or more of the replicas have been lost due to a system failure. A regeneration-based replica control protocol begins with a fixed number of replicas on various sites. As these replicas fail, they are replaced by new replicas that are created on additional sites called spares. Once a failed site is repaired, the regenerated replicas become superfluous and the additional storage used can be reclaimed. The question of which replicas should be reclaimed remains an area of future research.

With five or more participating sites and a high rate of regeneration, replacing a single replica with a spare has been shown to have only a slight detrimental effect on the reliability of the data object. As illustrated by Figure 5, the reliability decreases drastically if fewer replicas are maintained. Thresholds at which similar degradation occurs can also be observed for the other protocols that were considered.

There are costs associated with regeneration-based replica control protocols as well. Perhaps the most important is an increase in network message traffic. When a site fails, or the communication network becomes partitioned, a regeneration will occur. In order to generate a new replica, a copy of an extant replica must be transmitted. The cost of this may be prohibitive for large data objects such as data bases. For this reason, regeneration may be best suited for small data objects with strong fault tolerance requirements. Several ways of mitigating the cost of regeneration have been proposed [14].

Available copy protocols have been shown to provide the highest possible reliability for all replica control protocols studied, and these parallel those obtained for availability [15]. Using standard Markovian assumptions, closed-form expressions have been derived for the reliability of the tractible systems. The simulations of systems employing sites with non-Markovian distributions show that the Markovian models upon which the numerical solutions are based are indeed robust enough to predict the behavior of non-idealized systems.

The numerical solutions of the Markov models, backed by simulation results, established a hierarchy of protocols ordered by increasing reliability. They clearly indicate that the Available Copy protocol provides much higher reliabilities than the quorum-based protocols, and establish Dynamic-linear Voting as the protocol of choice for a communications network susceptible to partitioning. With estimates of the mean

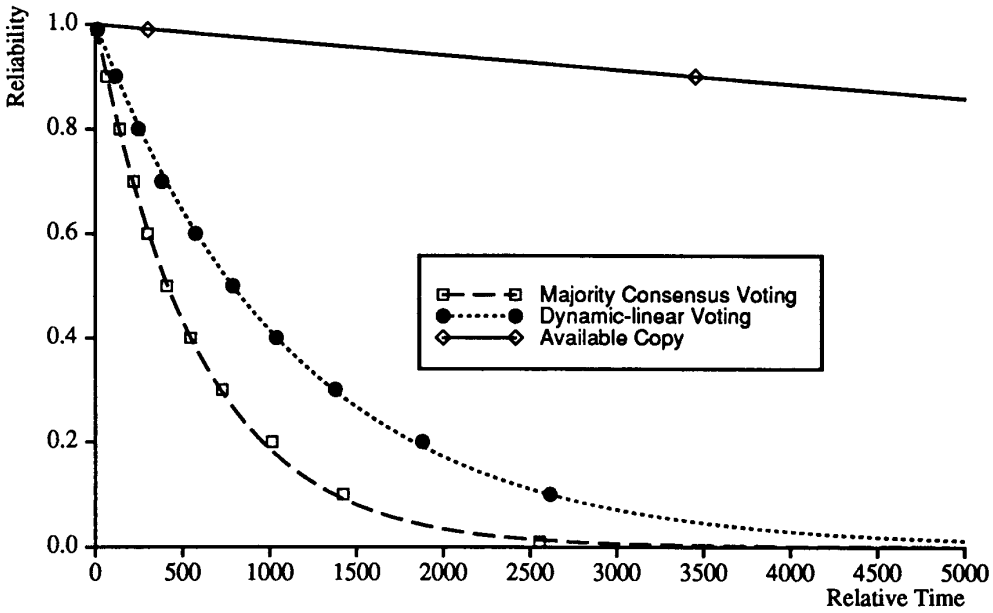

Figure 4: Compared Reliability of AC, DLV and MCV: $\kappa = 10.0, \lambda = 0.1, \mu = 1.0$

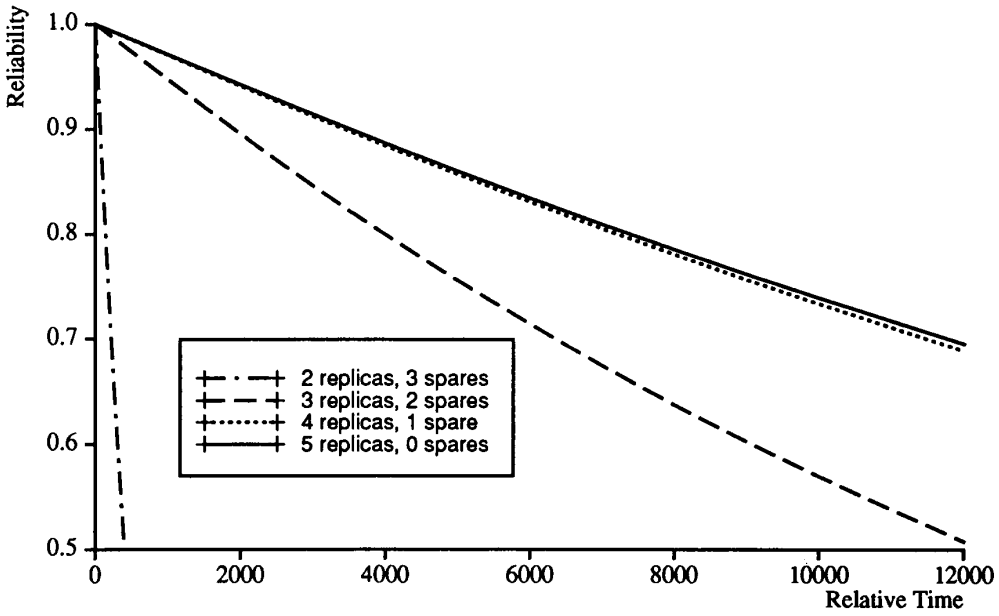

Figure 5: Compared Reliability for AC with Varying Numbers of Spares: $\kappa = 10.0, \lambda = 0.1, \mu = 1.0$

times to site failure and repair, the numerical techniques presented here can be applied to predict the reliability afforded by differing apportionments of sites and spares. Designers may in this way determine the fewest number of replicas which can provide the desired level of reliability.

Acknowledgements

The authors are grateful to Jehan-François Pâris, Alexander Glockner and Mary Long for their assistance. The closed-form solutions were found with the aid of *Maple*, a symbolic algebra program developed by the Symbolic Computation Group at the University of Waterloo. The numerical results were computed using MATLAB from MathWorks, Incorporated of Sherborn, Massachusetts. The simulation results were obtained with the aid of SIMSCRIPT II.5, a simulation language developed and supported by CACI Products Company of La Jolla, California.

References

[1] P.A. Bernstein and N. Goodman. "An Algorithm for Concurrency Control and Recovery in Replicated Distributed Databases." *ACM Transactions on Database Systems* 9 (December 1984): pp. 596–615.

[2] P.A. Bernstein, V. Hadzilacos and N. Goodman. *Concurrency Control and Recovery in Database Systems.* Reading, Massachusetts: Addison-Wesley, 1987.

[3] J.L. Carroll, D.D.E. Long and J.-F. Pâris. "Block-Level Consistency of Replicated Files," *Proceedings of the 7th International Conference on Distributed Computing Systems,* Berlin: IEEE, September 1987, pp. 146–153.

[4] D. Davčev and W.A. Burkhard. "Consistency and Recovery Control for Replicated Files." *Proceedings of the 10th ACM Symposium on Operating Systems Principles,* Orcas Island: ACM, 1985, 87–96.

[5] G.E. Forsythe, M.A. Malcolm and C.B. Moler. *Computer Methods for Mathematical Computations,* Englewood Cliffs: Prentice Hall, 1977.

[6] D.K. Gifford. "Weighted Voting for Replicated Data." *Proceedings of the 7th ACM Symposium on Operating System Principles,* Pacific Grove: ACM, 1979, pp. 150–161.

[7] G.H. Golub and C.F. Van Loan. *Matrix Computations,* Baltimore: The Johns Hopkins University Press, 1983.

[8] I.N. Herstein. *Topics in Algebra,* Blaidell Publishing Company, 1964, pg. 209.

[9] S. Jajodia and D. Mutchler. "Dynamic Voting." *ACM SIGMOD International Conference on Data Management,* ACM, 1987, pp. 227–238.

[10] D.D.E. Long. "The Management of Replication in a Distributed System." Ph.D. dissertation, Department of Computer Science and Engineering, University of California at San Diego, 1988.

[11] D.D.E. Long, J.L. Carroll and K. Stewart. "The Reliability of Regeneration-Based Replica Control Protocols," Technical Report UCSC-CRL-88-18, University of California, Santa Cruz, 1988.

[12] D.D.E. Long and J.-F. Pâris. "On Improving the Availability of Replicated Files," *Proceedings of the 6th Symposium on Reliable Distributed Systems,* Williamsburg: IEEE, 1987, pp. 77–83.

[13] D.D.E. Long and J.-F. Pâris. "A Realistic Evaluation of Optimistic Dynamic Voting," *Proceedings of the 7th Symposium on Reliable Distributed Systems,* Columbus: IEEE, 1988, pp. 129–137.

[14] D.D.E. Long and J.-F. Pâris. "Regeneration Protocols for Replicated Objects," *Proceedings of the 5th International Conference on Data Engineering,* Los Angeles: IEEE, 1989, pp. 538–545.

[15] D.D.E. Long, J.-F Pâris, and J.L. Carroll. "Reliability of Replicated Data Objects," *Proceedings of the 8th International Conference on Computers and Communications,* Phoenix: IEEE, 1989, to appear.

[16] C.B. Moler and C.F. Van Loan. "Nineteen Dubious Ways to Compute the Exponential of a Matrix," *SIAM Review* 20, 1978, pp. 801–836.

[17] J.D. Noe and A. Andreassian. "Effectiveness of Replication in Distributed Computing Networks." *Proceedings of the 7th International Conference on Distributed Computing Systems,* Berlin: IEEE, 1987, pp. 508–513.

[18] J.-F. Pâris. "Efficient Management of Replicated Data." *Proceedings of the International Conference on Database Theory,* Bruges, Belgium: 1988.

[19] C. Pu. "Replication and Nested Transactions in the Eden Distributed System." Ph.D. dissertation, Department of Computer Science, University of Washington, 1986.

[20] C. Pu, J.D. Noe and A. Proudfoot. "Regeneration of Replicated Objects: A Technique and its Eden Implementation." *Proceedings of the 2nd International Conference on Data Engineering,* Los Angeles: IEEE, 1986, pp. 175–187.

[21] L.H. Shampine and C.W. Gear. "A User's View of Solving Stiff Ordinary Differential Equations." *SIAM Review,* 21, 1979, pp. 1–17.

[22] D. Skeen. "A Quorum-Based Commit Protocol." *Proceedings of the 6th Berkeley Workshop on Distributed Data Management and Computer Networks,* Berkeley: University of California, 1982, pp. 69–80.