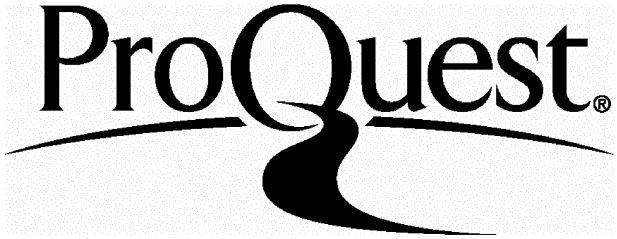
An Analysis of the Impact of Network Configuration on the Reliability of Distributed Systems

Iris Lay
Department of Computer Science,
University College London,
Gower Street, London WC1E 6BT
United Kingdom

A thesis submitted for the degree of Doctor of Philosophy
April 1998



Abstract

One of the advantages of distributed systems is their capability to improve the accessibility of data. Fault tolerance of data is improved by replication, where data is stored redundantly at multiple sites. This increases the likelihood that the data remains accessible in spite of site and communication failures. As networks grow and evolve, parts of the network can become separated form each other into sub-networks by link communication failure for example. This creates a situation where machines in one such sub-network are able to communicate with each other, but not with machines in other parts of the network. This peculiarity called partitioning may decrease the accessibility of data. There are two techniques for measuring the accessibility of data in a distributed system: availability and reliability. Availability has received much more attention the than reliability because reliability is much more difficult and impractical to analyse. Availability of a system is important, however, there are applications where reliability is a more important measure of a system's behaviour. Since reliability is a measure of continuous availability there may be situations where a system has high availability but be quite unreliable. This dissertation describes the development of a comparative reliability measure that provides a satisfactory and practical technique for the comparison of replication control algorithms. This measure is used to investigate the effect of the network topology, consistency control techniques and copy placements on the reliability. A simple notation is developed and is used to illustrate the presentation of large amounts of comparative reliability data in an easily assimilated form which can be used to conduct performance investigation to aid analysis of how different replication techniques are affected by network configuration and copy placement.

Acknowledgments

I am indebted to Dr. Søren-Askel Sørenson, for coming to my aid in finishing this work. His encouragement and support has been invaluable.

I wish to thank Mr. Benjamin Bacarisse for his assistance and supervision during the course of my research. His enthusiasm, disposition and patience were a constant source of inspiration. I wish to take this opportunity to wish him all the best of health and life.

I am grateful to the Science and Research Council and the Computer Science Department at the University College London for providing me with the opportunity to complete this work.

I would like to make a special note of thanks to my husband for his support during the research and completion of this work.

Table of Contents

Chapter 1

Introduction

Distributed systems consist of a number of independent processing units (nodes) connected by a communication network which closely interact in order to fulfill an overall goal. In theory, the nodes in a distributed system are transparent to the user, but in reality very few systems accomplish this strictly [105]. The potential benefits of Distributed Systems include their capability to improve the accessibility of stored objects by the use of replication, where these objects are stored redundantly at multiple sites. Replication increases the likelihood that these objects remain accessible in spite of site and communication failures. It also improves the response time and throughput when storage of replicas and processing are done locally. Bottlenecks may also be reduced by the parallelism of multiple processors.

As networks grow and evolve, parts of the network can become separated from each other into sub-networks by link communication failure for example, and a situation occurs where machines in one such sub-network are able to communicate with each other, but not with machines in other parts of the network. This peculiarity, called partitioning, where functioning sites in a distributed system are unable to communicate, is the most disruptive of all possible communication failures in a distributed system. The integrity of replicated objects may be compromised if, during a partition in a distributed system, several independently located updates are being performed on the same object in separate sub -networks that cannot communicate and are not aware of the conflict. Fortunately, there are replication control techniques that allow partitioning, however, the presence of partitioning, may bring about a decrease in the accessibility of replicated objects. The extent of the decrease in the accessibility of replicated objects due to partitioning has not been quantified in the literature, this work provides techniques to enable the comparison of partitioned networks and non-partitioned networks and the decrease in the accessibility of replicated object when partitioning is introduced.

There are two techniques for measuring the accessibility of data in a distributed system: availability and reliability.

- Availability (usually denoted by A) is a **single** value, which makes it simple to use when comparing systems. It can be defined as the probability that an object is accessible at any given moment. This definition can be summarised as the total portions of time that an object is accessible for any given time interval. Availability has received much more attention than the reliability because it is easier to analyse and manipulate as a measure. (See section 2.6.1).

- Reliability R(∞t) on the other hand, is more difficult to manipulate, it is a **function of time intervals** rather than a single value. It can be defined as the probability that an object will be continuously available during an arbitrary time interval, ®t. This definition can be summarised as the total time portions that a replicated object is continuously available for during any given time interval. Analytical evaluation is therefore complicated if not impossible in some cases. Manipulating the reliability as a measure in comparing systems is more demanding than the manipulation of the availability. (See section 2.6.2)

The availability of a system is important. However, there are applications where reliability is a more important measure of a system's behaviour. Consider an object that fails frequently for short periods as opposed to one that fails rarely, but is accessible, on average for a lower proportion of the time. The availability for both objects would remain the same, but the reliability would be considerably different. There may be situations where a system may have high availability but be quite unreliable, since reliability is a measure of **continuous availability**. Availability therefore, can at least be inferred from the reliability if only it were practical to manipulate. Reliability calculations may be complicate further by including the impact of the replication control algorithm, network configuration and copy placements. The impracticality of calculating the reliability in this way, has inspired the development of a new and promising approach to the measurement of accessibility: **comparative reliability**.

The comparative reliability measure is used to investigate the resiliency of replication control techniques to changes in network configuration and the placement of copies. Questions such as:

- How do replication control techniques impact the reliability
- How does the network configuration impact the reliability
- How does the placement of copies impact the reliability

can be answered with the utilization of the comparative reliability measure developed in this Thesis.

1.1 THE THESIS

A comparative reliability measure is developed that provides a satisfactory and practical technique for the comparison of replication control algorithms. This measure is used for investigating of the effect of network configuration, consistency control techniques and copy placements on reliability. A simple notation technique is developed that enables the presentation of large amounts of comparative reliability data in an easily assimilated form.

1.1.1 OBJECTIVES

Find Comparative Reliability Measures

The first objective of this thesis is to find comparative reliability measures that capture all the significant information of the reliability function in one value.

Determine a Suitable Notation

The second objective is to determine a suitable notation technique that represents clearly and concisely all the numerous parameters and results that are involved in a comparative study of different network configurations, control algorithms and copy placements.

Select an Appropriate Investigation Technique

The third objective is to find a practical technique for comprehensively investigating the resiliency of replication algorithms to changes in network configuration and copy placement.

Result Analsysis

The fourth objective is to combine the work on reliability measures, notation techniques and the results of the investigation to draw constructive conclusions about the impact of the network configuration and copy placements on the reliability of Distributed Systems.

1.1.2 OUTLINE

A chapter 2 comprises critical and background information relevant to this thesis. It includes a presentation of a model of distributed systems and their failure types. Replication is also introduced followed by an outline of the consistency and concurrency control problem for replicated data. Finally the management of replicated data is discussed.

Chapter 3 contains a general survey of replication control techniques and provides an explanations to the algorithms that were chosen for the investigation and why.

Chapter 4 describes measures of comparative reliability and the presentation technique used to display the numerous parameters involved in the investigation.

Chapter 5 outlines three distinct approaches for the examination of reliability and explains the reason for choosing simulation experiments for this investigation.

Chapter 6 describes the simulation environment, the experimental procedure and the parameters of the simulation. This is followed by the validation of the experiments and an experimental plan. Differen t network topologies are introduced from which any arbitrary network configuration can be constructed. These topologies are manipulated during the experimentation stages to investigate the resiliency of replication algorithms.

Chapter 7 provides a summary of the results of the experiments. A comparative reliability measure is used to compare replication algorithms implemented in different network configurations with a comprehensive study of copy placements.

Chapter 8 concludes this Thesis and provides suggestions for future work.

Chapter 2

Background

The growth in microelectronics has brought about generally cheaper hardware. This combined with similar cost reductions and technological advances in communications have, in part, been the cause for the migration to Distributed Systems. User demand for faster, more accessible and more sophisticated systems has been a further motive for the progress into Distributed Systems.

2.1 DISTRIBUTED SYSTEMS

A Distributed system consists of a number of processing units (nodes) connected by a communication network, which closely interact in order to fulfill an overall goal [59]. With a system that is not distributed, a user must explicitly log onto a particular machine, explicitly move files around, and generally, all the network management must be handled personally. In a distributed environment, system-wide management and control is necessary. A refined definition of a distributed system is given in [105]

'A distributed system is composed of a number of autonomous processors and/or data stores supporting processes and/or data bases which interact in order to cooperate to achieve a common goal. The processes coordinate their activities and exchange information by means of information transferred over a communications network.'

Objects in a distributed system may be distributed over several nodes. Therefore, manipulating objects that are stored in a distributed system may involve accessing a multiple number of nodes. Difficulties one may encounter in accessing objects can be magnified, since the probability of success, is related to the product of the probabilities that the accessed nodes are operational. Replication is an approach that can be utilized

to increase the probability of success. This can be done by placing copies of critical items on separate nodes with independent crash probabilities.

Replicated objects in Distributed Systems are not limited to data files or databases, they may include directory structures, essential system files, or any storage objects that require faster or greater accessibility. However, it is important to understand the concept of a Distributed File System, which may then be applied to other storage objects.

2.1.1 DISTRIBUTED FILE SYSTEM

In order to appreciate the structure of a Distributed File System (DFS) we need the following definitions:

Service A software entity running on one or more machines and providing a specific type of function to any client requesting that service.

Server The service software running on a single machine.

Client A process that can invoke a service using a set of operations that form its client interface.

Distributed File System (DFS)

> A distributed implementation of a file system can be described as a system file system where multiple users share files and storage resources. It is a file system whose main purpose is to provide long-term storage, and whose clients, servers and storage devices are dispersed among the machines of a distributed system. Service activity has to be carried out across the network, and rather than using a single centralized data repository, multiple and independent storage devices are used [81].

The concrete configuration and implementation of a DFS may vary. There are configurations where servers run on dedicated machines, as well as configurations

where machines can act both as servers and as clients. A DFS can be implemented as part of a distributed operating system or, alternatively, by a software layer whose task is to manage the communication between conventional operating systems and file systems.

The performance of a DFS can be improved by replicating copies of shared data on processors where they are frequently accessed, this reduces the need for expensive, remote accesses, since an application can use neighbouring copies of the data instead of distant ones.

2.2 THE DISTRIBUTED SYSTEM MODEL

Since failures due to partitioning are the main focus of this work, a distributed system is modeled as a graph of subnets rather than of nodes to keep the terminology consistent with other work in this area.

Definition 2.1 Let S_i represent a fully connected set of nodes $N = n_1,....,n_k$ such that the communication links connecting these nodes never fail. This set is called a subnet.

Nodes represent the machines in a distributed system. Node failures are not correlated, this means that if a node fails it does not influence any other nodes to fail.

Definition 2.2 Let G(S,L) represent a distributed system as a probabilistic graph of a collection of subnets, interconnected by a set of edges representing the communication of bi-directional links, where S={1,2,...,n} is the set of subnets and $L=(s_1 , s_2),(s_2, s_3),...,(s_{n-1} , s_n)$ is the set of links.

This definition does not limit the investigation since single nodes can be represented by subnets containing one node. The link can represent a bridge, a router, a gateway, or a server that behaves as both server and bridge.

2.2.1 FAILURE MODEL

Failures in the distribute system can arise in a number of different ways:

1. Node crashes. These are random occurrences where the failure of one node does not affect the failure of other nodes.

2. Communication link failure, which is also a random occurrence where the failure of one link does not affect the failure of other links. However, they can cause partitioning failures (see section 2.3).

3. Partial communication failure where a bi-directional link can communicate in one direction but not another.

4. Action by intentional enemies, there are purposeful agents who selectively inflict damage to render the network non-operational.

5. Natural catastrophes, such as earthquakes and hurricanes. These failures are not purposeful; nevertheless, they typically damage some portion of the topology in a small geographical region.

6. Byzantine failure, which consists of software failures that, cause the system to operate incorrectly even when the hardware does not fail [70]. For example the flow control algorithm causing the network to be flooded with traffic, resulting in failure due to overload, or the routing algorithm's inability to detect a functional route, even though one exists.

In this work, nodes are fail-stop [99] i.e., the only failure they have is a halting failure which means they stop processing.

A fault tolerant system [17,18,42,63,73,92,104,114] should continue functioning, perhaps in a degraded form in the face of failures, which broadly includes

communication faults, node failures (of type fail stop), storage device crashes, and decays of storage media. The degradation can be in performance, functionality, or both and should be proportional, in some sense to the failures causing it.

2.3 NETWORK PARTITIONING

The most disruptive of possible communication failures in a distributed system are network partitions. These are failures that separate the network into isolated subnets called *partitions.* In a partition, nodes can communicate with each other but cannot communicate with nodes in other subnets.

When partitioning occurs, nodes in one partition might perform an update on a file while nodes in another partition perform a different update on the same file. If these two updates conflict, it may be difficult or impossible to resolve the conflict satisfactorily.

The design of replication techniques that tolerate partitioning is remarkably demanding. Typically, a node in a particular subnet cannot distinguish between cases where other nodes are simply isolated from it and cases where the nodes are down or not functional. At best, it may be able to identify the other nodes in its partition. Evaluating the cause and extent of a partition is therefore practically impossible.

Slow responses from some nodes can give the impression that the network is partitioned when it is not. Furthermore, in practice, communication between two subnets may be possible in one direction but not the other. This results in an interesting situation, one which can be described as semi-partitioned, since updates can occur in a partition that may accept messages, but where its own messages will not be received.

Definition 2.3 A path P_{pq} between a given pair of subnets s_p and s_q is a set of edges linking them together such that $P_{pq} = (s_p, s_1), (s_1, s_2), \ldots, (s_{n-1}, s_n)(s_n, s_q)$.

Definition 2.4 A route exists between any pair of subnets s_p and s_q if there is a path P_{pq} between them such that all the links are up.

The assumption is that if a route exists between any two nodes then the distributed system is capable of finding it.

Definition 2.5 Let G(S,L) represent a distributed system. A partition occurs when there is no route from subnet $s_i \in S$ to any other subnet $s_j \in S$, but the nodes belonging to that subnet are working (up).

Many authors have stressed the importance of partitioned operation in distributed data management (e.g. [6,13,28,31,37,50,106]). However, there is little information on the frequency of partitioning in actual systems, or information on case studies of particular systems. There is no theoretical information or estimation of how often partitioning can be expected, whether it is a frequent or rare occurrence (and if so how frequent or rare), whether partitioning is caused by failures or is the result of anticipated events.

There are two extreme strategies for dealing with partition processing largely depending on the particular application and policy of a specific system; pessimistic and optimistic techniques (see section 2.6).

2.4 REPLICATION

Definition 2.6 An object in a distributed system is a data structure, typically replicated among the address spaces of multiple processes. Each object has a type, which defines the set of possible values the object can assume. Each object also has a set of operations that provide the only means to access or modify the object. An operation is split into two parts, namely, an invocation sent by a process to an object, and a response sent from the object to the process at a later time.

Replication increases availability and reliability of data. Formal definitions of these will be given later, however, intuitively, replicating an object on nodes with independent failure modes, increases the availability of that object, i.e. the probability that at least one replica will be accessible increases.

Consider a replicated data file, if one takes an operation such as a read on this file it is important to make the distinction that availability and reliability are properties of the operation rather than of the data.

2.4.1 CONSISTENCY CONTROL

When using replication techniques, one must make the collection of replicas of an object behave as if there was one single data structure, thus enabling the user to only see the most recent version of an object [26,57]. Essentially this means being able to control the operations of an object in a manner that presents inconsistencies among the different replicas due either to the effects of competing operations or to the effects of crashes during an operation.

For instance if two clients open the same file for update, and then each of them simultaneously issues a request to replace the first record of the file, the request that happens to be serviced last will overwrite the first update. This phenomenon is termed in the literature as serial consistency or one-copy serializability [14,34].

2.4.2 CONCURRENCY CONTROL

Atomic transactions that either succeed or fail without disturbing the present version of an object are required in order to preserve serial consistency [22,98,110,111,112,]. This is sometimes termed the "all or nothing" property. Mechanisms for implementing atomicity (concurrency control techniques) fall into several broad categories, depending on how the serialization order for transactions is chosen. The serialization order may be predetermined, as in multi-version time-stamping schemes (e.g.[16,55,86,109]), or it may be chosen dynamically, as in two-phase locking schemes (e.g. [33,67,83]). There are also hybrid techniques employing both locking and methods resembling time-stamping (e.g.[69,85]). Means for update crash recoveries are also well documented (e.g. [41,45]).

2.5 REPLICATION STRATEGIES

There are two types of replication strategies. This section describes and compares these strategies.

2.5.1 PESSIMISTIC REPLICATION TECHNIQUES

Pessimistic techniques function by limiting availability and ensuring consistency. The underlying pessimistic assumption is that if an inconsistency can occur, it will occur. Therefore although pessimistic techniques differ in how they ensure consistency, ultimately, consistency is guaranteed under partitioning by restricting the transaction processing [e.g. 1,5,8,9,12,19,36,38,43,61,78,87]. Several of these techniques will be described in Chapter 3.

2.5.2 OPTIMISTIC REPLICATION TECHNIQUES

A consistency control technique is optimistic if it allows transactions to execute without synchronization, relying on commit-time validation to ensure serializability. These techniques therefore do not limit availability. A transaction will be executed in any partition that contains the right number of copies in order to comply with that particular replication technique protocol. Hence, although transaction processing within each partition is consistent and no user operating within a single partition would detect an inconsistency, global inconsistencies can occur. Once the system is reconnected, it must first detect inconsistencies and then resolve them [27,43,53,54].

2.5.3 PESSIMISTIC VS OPTIMISTIC TECHNIQUES

The advantage of optimistic techniques is obviously higher availability. They achieve this by minimizing lost opportunity, which is the cost associated with needlessly delaying a transaction. These costs may be significant for certain application especially were user satisfaction is important. However, lost opportunity may still occur in optimistic techniques due to the delays caused by repairing conflicts.

he cost of an optimistic technique is the overhead of conflict detection plus the repair ost, whereas the cost of a pessimistic technique is the cost of opportunities lost to real nd apparent partitioning [27].

optimistic techniques are to be useful in general-purpose systems, it must be possible apply them selectively in conjunction with appropriate pessimistic techniques.

he comparison of any two different pessimistic techniques is a lot less complicated an the comparison of any two optimistic techniques. Optimistic techniques basically y to achieve the same thing, i.e. faultless data consistency. However, the optimistic echniques have varying degrees of data consistency, which complicates any means by hich a comparison can be made. There would obviously be an explosion of additional ariables and parameters to consider (unless you could be sure that two optimistic ethods are trying to achieve the same degree of consistency). After careful onsideration, only pessimistic techniques have been chosen for this dissertation.

.6 MEASURES OF ACCESSIBILITY

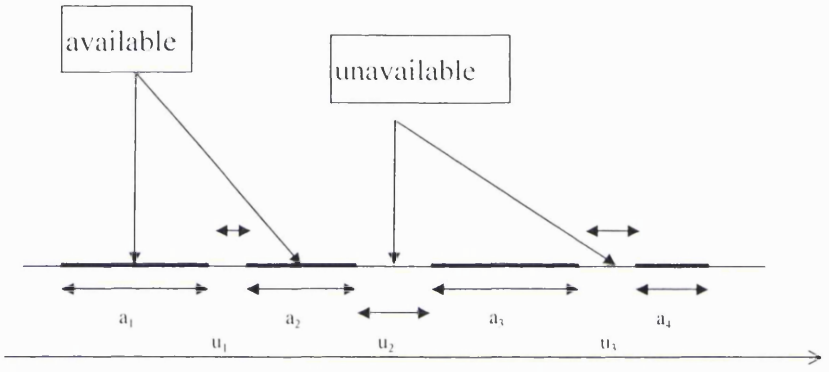
non-replicated object is unavailable during the period that it takes for its holding node crash and recover. For example a file may not be updated and is not accessible even r reading until the file-server recovers.

here is no doubt that replication techniques increase the accessibility of an operation n some object, however, the accessibility due to replication has to be quantified not nly as a guide to system managers, but for comparison purposes. Managers have to alance the cost of maintaining and managing replicated copies against the increase in eliability and performance in their systems.

.6.1 AVAILABILITY

he simplest measure of accessibility is availability. Consider Fig 2.1 in which the top ne represents an object over a period in time. The bold line represents periods in

which it is operating correctly, the fine line represents periods in which it s not. Availability of the object is the long-term proportion of the bold line.



Availability (usually denoted by A) is a **single** value, which makes it simple to use when comparing systems. It can be defined as the probability that an object is accessible at any given moment. This definition can be summarised as the total portions of time that an object is accessible for any given time interval.

Definition 2.7 An object alternates between being accessible and being inaccessible as a result of failures (f_i) and recoveries (r_i) at times $r_0 = 0$, $f_0 > r_0$, $f_1 > r_1$, ... Let

$$U(t) = \sum_{f_i \le t} (f_i - r_i)$$

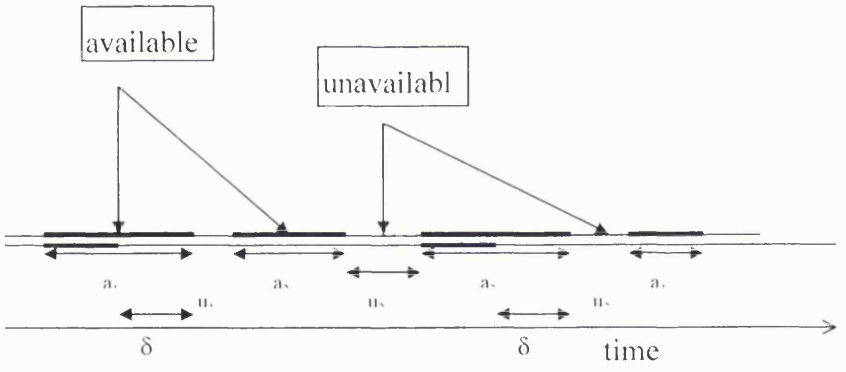
be the total time the object is accessible from time 0 to time t. For any finite t we may define the availability A as

$$\lim_{t \to \infty} \frac{U(t)}{t}$$

Consider a replicated file whose operations are read and write. Availability for a read transaction (read-availability) A_r , may be different to that of a write transaction (write-availability) A_w . For example, using a consistency control technique often called the "read one write all", the read-availability will be higher than the write availability because the file will be available for reading as long as one copy is available, but will only be available for writing when all copies are available.

2.6.2 RELIABILITY

Consider Figure 2.2, in which the bold portions of the bottom line represent the periods of time that a replicated object will be continuously available for the next δ time units. The reliability $R(\delta)$ of the object is the log-term proportion of these bold lines.



Reliability R(®t) is a **function of time intervals** rather than a single value. It can be defined as the probability that an object will be continuously available during an arbitrary time interval, ®t. This definition can be summarized as the total time portions that a replicated object is continuously available for during any given time interval.

Definition 2.8 Let reliability

$$R(\partial t) = \lim_{t \to \infty} \frac{U(\partial t, t)}{t}$$

where

$$U(\partial t, t) = \sum_{f_i \le t} H(f_i - r_i - \partial t)$$

and H(x) is the continuous Heavyside function: H(x) = x for x >0 and 0 otherwise.

Availability has received much more attention than reliability. This is partly because the analysis of availability is easier to attain than that of reliability. In fact, for any complex systems, reliability is difficult to calculate analytically [24,39,46,48,58,60,82,84,93,94,106]. Even for seemingly simple cases the analytical calculations are extremely arduous (see section 5.1.3 for reliability of a pair of notes connected by a double link network). There are applications, however, such as process control, data gathering and tasks that require interaction with real-time processing where data will be lost if not captured when it is available. For these, reliability of the system is a more important measure of its performance than its availability. Consider an example of a phone line that failed for about a day in every year A=0.997. The probability of being able to make an uninterrupted hour-long call works out at 0.997. If it takes approximately 10 seconds to dial a call, a line that dropped the connection about once every hour (requiring a re-dial) would have the same availability, but the 1 hour reliability

(R(1 hr)=0.371) is 0.371 and this means that there is only a 37.1% chance of making an uninterrupted hour-long call. This example is not a "real" model, since telephone systems are designed for reliability not availability, however it does make the point.

Reliability is a **function of time intervals.** Therefore, it is more difficult to use than availability when making comparisons between systems. Chapter 4 describes various possible measures of comparative reliability and proposes the new comparative reliability measure briefly mentioned in the Introduction (Chapter 1). This comparative reliability measure is used extensively in this thesis to compare the reliability of different network configurations.

Reliability has been studied in the engineering literature [10,35,40,44,71,74], however, the emphasis in the engineering literature has been on analysing the distribution of times to first failure.

2.7 MANAGEMENT OF REPLICATED DATA

Commercially available distributed file systems have tended to concentrate on the problems of providing efficient remote access, rather than offering increased reliability through replication [29]. Ironically, system administrators are then tempted to distribute functionality across the network, thereby decreasing the overall reliability of applications.

The increased complexity of systems that support replication should not be underestimated. This complexity can arise from the replication algorithms themselves or from extra support services that are required. If we consider the case of a general-purpose file system that supports replicated files, it becomes clear that replication will complicate many other areas of the system's design. System administrators have to consider additional problems, for example:

- Which replication technique is most suitable for their particular needs?
- Which files should be replicated and which should not [97]?

- How should disc usage quotas be determined or enforced [66]?
- How does replication affect the security of files [2,52,98]?
- Where in the network system to put replicas [20]?
- Who should decide; the user or the system administrator?
- How transparent should replicas be [103]?

In some sense the users should not really be concerned with whether a file is replicated or not, therefore the existence of replicas should be invisible at higher levels of a system. At some level, however, the replicas must be distinguished from one another by having different lower level names.

Under certain circumstances, it is desirable to expose these details to users. LOCUS, for instance [88], provides users and system administrators with mechanisms to control the replication scheme.

Levy and Silberschatz [72] have surveyed some systems that use replication excellently. File systems such as Coda and Echo provide read-write replication of data. Amoeba supports read-write replication at the directory level because files are immutable in that system. Although read-write replication is well understood theoretically, as yet, very little practical experience of its uses exists. More experience has been gathered with read-only data replication, which is supported by systems such as Sun NFS and Andrew. Though suitable only for files that change relatively rarely, it is valuable because many critical files (such as system binaries) possess this property.

In coming to a decision about which replication algorithm to implement, system administrators need to consider the following points:

1. **Practicality**. Some algorithms (e.g. Dynamic Voting [61]) assume that every node will become aware of the failure of any other node as soon as

it has occurred. Implementing this requirement can be very difficult in practice.

2. **Resiliency**. How resilient is the algorithm to changes in the network configuration and to the placement of copies.

3. **Storage Cost**. The number of copies needed before there is an improvement in availability and reliability. For example weighted voting needs a minimum of three copies in order to improve availability of a file.

4. **Fault Tolerance**. For example, does the algorithm tolerate communication failures that cause partitioning.

5. **Replication Strategy**. Is the algorithm pessimistic, or optimistic? If it is optimistic, what is the degree of consistency and how costly is the detection and resolution of inconsistencies.

Research in distributed systems has been criticized for devising strategies for isolated problems [90]. In particular, there are no helpful guidelines for system administrators with practical information of how different replication techniques are affected by the network configuration and copy placement, which would help make some of these decisions.

This study endeavors to resolve some of these problems by providing a measure of reliability, which administrators and researchers can use to conduct performance investigations. This will help in making the decision as to which replication algorithm is more suitable for their requirements.

Chapter 3

Consistency Control Techniques

This chapter describes a selection of pessimistic replication control techniques. Pessimistic rather than optimistic consistency control techniques were selected for this thesis to allow for an equitable comparison.

Pessimistic techniques are basically striving to achieve the same thing, i.e., faultless data consistency. In contrast, optimistic techniques offer varying degrees of data consistency. They generate many more additional variables and parameters to consider and still one cannot be sure that any two optimistic techniques are aiming to achieve the same degree of consistency. Although optimistic consistency control techniques are very interesting, for the purposes of this thesis, pessimistic techniques offer the basis for an acceptable comparison of replication techniques.

3.1 UNANIMOUS UPDATE

Unanimous Update is a basic approach to replication that requires all copies to be identical before and after each operation. A "read" operation is allowed to execute on any copy but a "write" operations must update all of the copies or none at all; in which case the write operation fails.

This technique increases the availability for read transactions (read availability) compared with a single copy, however the availability for write or update transactions (write availability) decreases as the number of copies is increased. Furthermore, the system is required to

support control message traffic in order to send updates to all replicas and confirm or cancel the update based on whether or not the update was agreed by all the replicas. This means that the implementation of this algorithm requires a two-phase commit protocol or some form of locking mechanism for confirmation, in order to prevent inconsistencies.

Unanimous Update [16] does not tolerate machine failures for updates and hence offers very low reliability compared to all other techniques. However, it does offer consistency even in the face of network partitions by preventing updates in any partition that does not have access to all the copies. It is a technique that is most efficient for situations where read operations are far more common than write operations.

3.2 SINGLE PRIMARY UPDATE

In this scheme one copy is designated as primary and all the others as secondaries [3]. Update or write requests are sent to the primary copy, which then obtains a lock and performs the update. Once this has been accomplished the primary will broadcast the update to all the secondaries and releases the lock. Consistency is guaranteed by serializing the updates. Once the primary has been updated, there are three methods by which it broadcasts the updates to the secondaries:

1. it sends the update broadcast to the secondaries immediately;

2. updates to the secondaries are sent at the end of a particular transaction;

3. updates are broadcast only at specific intervals (e.g. once an hour, overnight, etc.).

For all replication algorithms based on the primary/secondary process there is obviously an increase in the response time for read transactions that follow an update because the primary must propagate updates to the secondaries. The increase in response would be especially apparent if the secondary is local and the primary is remote.

The main disadvantage of this approach is the fact that it does not tolerate the failure of the primary copy. However, it does maintain consistency in the face of network partitions. Since only the partition containing the primary copy can access the data. On recovery, updates are sent to the secondaries in order to preserve consistency.

The write availability for this method corresponds to the availability for a single copy, but the read availability is increased.

The simplicity of this technique has made it popular with many practical designs [62,85].

3.3 MOVING PRIMARY UPDATE

Alsberg [3] proposed an extension to the Single Primary Update algorithm. He extended the concept by allowing an update transaction to be made to the primary copy or any secondary copy. There are two different circumstances in which an update may be requested:

1. The primary receives the update, performs the update and then sends a co-operation request to one of the secondaries informing it of the update. The secondary performs the update, acknowledges the primary and passes the request on to another secondary. An update is accepted only once the acknowledgment is received by the primary, and two host resiliency has been achieved. The update is rejected only if both primary and the co-operating secondary fail.

2. The secondary receives the update and forwards the request to the primary. The primary then proceeds as in (1).

If the secondaries discover that the primary has failed, they elect a new primary among themselves. In a two-host, resilient scheme, all *n-1* secondaries, where *n* is the number of

copies, must participate in this election. When the old primary recovers, and attempts to ask for co-operation for an update, it is informed by the secondary of its replacement, and the request is forwarded to the new primary. The old primary then becomes a secondary.

In general, in an *m*-host resilient scheme at least *n-m+1* secondaries must participate in the election of the primary. The rest of the algorithm is the same as the two-host case above. Garcia-Molina discusses different election protocols in [37].

To obtain the best results with this algorithm, machine failures must be distinguishable from network failures. If there is uncertainty as to whether the failure is due to a machine failure or to a network partition, the assumption must be that there has been a partition and no new primary can be elected.

The write availability is only increased if there are more than two copies. In the two-copy case, both copies are required for updates; one as the primary and one as the co-operating secondary, so the algorithm behaves like a Unanimous Update.

Since it is extremely difficult in practice to distinguish between node failure and network partitions this technique is rarely used.

3.4 AVAILABLE COPIES

The available copies algorithm [16] is an extension of the unanimous update approach, in that rather than writing to all copies, update or write transactions are only sent to available copies, i.e. copies held on nodes that have not failed.

When a node fails, the failure is automatically detected, and the node is configured out of the system; no further operation will execute at that node until it is repaired.

Once a node is repaired, each available node must be checked to see if it was updated during

the absence of the failed node. If not, the copy may be used. If an update has occurred, the returning copy must bring itself up-to-date by copying the data from other available nodes before accepting any client transactions. If a copy returns upon node recovery and no other copies are found, then a manual update procedure has to be resorted to because a later version may have existed but all nodes might have failed in the interim.

Read transactions may execute on any available copy. Each copy maintains a directory list of available copies for use and the algorithm runs status transactions to keep these lists up to date as nodes fail and recover. A reliable, error free transport protocol is therefore required for the communication between nodes.

Since the algorithm assumes detection of node failures only, it is unable to distinguish between a failed node and one that is simply on the other side of a partition. Therefore, updates may be accepted by copies in more than one partition, thus leaving the system in an inconsistent state.

Adaptations of the available copies algorithm that do tolerate network partitions have been developed. For example:

1. Chan and Skeen [23] have developed an implementation of available copies that incorporates a method for detecting potential inconsistencies and calls for manual intervention.

2. El Abbadi et al. [30] extended the technique by defining a partition in which operations can be executed only if a majority of the copies reside there. The algorithm requires the implementation of an abstract communication layer on top of the real communication network, where the behaviour of the new layer approximates that of the network and uses virtual partitions which are rough analogs of the actual partitions that occur in the real network.

3. Bacarisse and Bek-Baydere's Reliable Histories (RH) algorithm [7] reduces the storage cost by replicating a small amount of information (histories) using a voting algorithm which can be used to implement a version of the available copy algorithm. When a

request arrives at a node, it consults with the histories in order to find the up-to-date file copies. Updates are only possible in a partition that includes a majority of history votes. RH provides high availability even for two copies.

Although the Available Copies algorithm cannot handle network partitions, it is described here because it offers optimal availability for pessimistic algorithms.

3.5 VOTING TECHNIQUES

The following section describes several of the most significant static voting techniques. In general these algorithms are robust, and remain consistent in the face of network partitions and node failures. In its simplest form, voting assumes that the current state of a replicated file is the state of the majority of its copies.

3.5.1 MAJORITY VOTING

With this technique, every copy has a number of votes associated with it. An operation can succeed only if it is applied to a set of copies that, together, hold a majority of votes. This ensures that every operation is performed on at least one copy that has participated in all previous successful operations. Each node maintains a version number so that the most up-to-date copy can be identified.

Read Operation

Votes and version numbers are collected until a majority is held, then the latest version is identified and read.

Write Operation

Votes and version numbers are collected and a pending write is sent to these nodes, until a majority of the votes is held. Once the node with the latest version number is

established, an attempt is made to write the file to a set of nodes that hold a majority of the votes.

If a majority of votes is not found the transaction is rejected and the file is considered unavailable.

For example consider a network with seven copies with their respective votes and assume the object has been updated 4 times as follows:

	A 3 Votes	B 2 Votes	C 2 Votes	D 5 Votes	E 1 Vote	F 8 Votes	G 6 Votes
Version No.	4	4	4	4	4	4	4

The total number of votes is 27. Now let us suppose the network has partitioned into two subnetworks and the user has requested an update.

	A 3 Votes	B 2 Votes	C 2 Votes	D 5 Votes	E 1 Vote	F 8 Votes	G 6 Votes
Version No.	4	4	4	5	5	5	5

The majority of the votes are in the partition containing nodes {D,E,F,G}. If the network partitions again as follows:

	A 3 Votes	B 2 Votes	C 2 Votes	D 5 Votes	E 1 Vote	F 8 Votes	G 6 Votes
Version No.	4	4	4	5	5	6	6

A majority of the votes can be found in the partition containing nodes {F,G}. Any further partitioning to this network would result in the update operation being rejected.

A minimum of three copies is required for increased availability. With two copies, either both copies are always needed in order to form a majority (which gives reduced availability compared with a single copy), or one of the copies is always required (which does not give an increase in availability compared with a single node).

3.5.2 WEIGHTED VOTING

The Majority Voting algorithm was extended by Gifford [38]. He made the algorithm more flexible by associating with each replicated file x a read quorum, $q_r[x] > 0$ and a write quorum, $q_w[x] > 0$ with the property that $q_r[x] + q_w[x] > n[x]$, where $n[x]$ is the total number of votes associated with file x. As in majority voting each copy has a version number associated with it.

Read Operation

is performed by accessing a set of copies of x that together hold at least $q_r[x]$ votes and reading the value associated with the copy having the highest version number.

Write Operation

is performed accessing a set of copies of x that together hold at least $q_w[x]$ votes and updating their version numbers to be greater than the maximum version number associated with any of those copies.

The property that $q_r[x] + q_w[x] > n[x]$ ensures that every read operation overlaps with every previous write operation. By controlling the write and read quorums, read availability can be

traded off against write availability and vice versa. For example, setting $q_r[x]=1$ and $q_w[x]=n[x]$ produces behaviour that is essentially unanimous update.

Gifford [38] shows how various combinations of vote assignments to the replicas and choices of $q_r[x]$ and $q_w[x]$ can be used to improve performance, for instance, copies on faster or neighbouring machines can be given more votes. Gifford also demonstrates how to optimize the system for particular read/write ratios.

3.5.3 GENERALISED QUORUM CONSENSUS

Even greater flexibility was introduced by Herlihy [51]. He describes a more configurable mechanism called generalized quorum consensus in which the semantics of the operations being performed are exploited to improve the availability even further. Since files are uninterpreted sequences of bytes, little is gained by applying the technique to files, but the method can be used to improve the availability of replicated directories [19]. Quorum consensus presents a general method for exploiting the type-specific properties of the data being replicated.

Generalized quorum consensus differs from the other voting algorithms in that the object's state is represented as a log containing all the modifications to that state, rather than just the state data themselves. Timestamps are used to order all the log entries.

3.6 VARIATIONS OF VOTING

Voting techniques are effective in providing high availability for larger numbers of replicas (5 or more). For large data objects the extra storage cost may not justify the increase in accessibility. There is also a trade off between the read availability and write availability.

In the following sections three variations of voting are described. The first reduces the storage cost, the second increases the write availability and the third increases the accessibility and so

can be used more effectively with only three copies.

3.6.1 VOTING WITH WITNESSES

The storage cost of the weighted voting algorithm is reduced by replacing some copies by entities called witnesses [87]. Witnesses are (simply) recordings of the version number of a replica with no data attached to it. Witnesses are assigned weights like conventional copies and participate in the collection of quorums. Whenever a witness is included in a write quorum, its version number is incremented every time the file is updated. The only obvious restriction is that each quorum must include at least one current copy, since you cannot read or update witnesses alone.

The availability provided by this technique for two copies and one witness is very close to the Weighted Voting technique availability with three full copies.

3.6.2 VOTING WITH GHOSTS

This approach, proposed by Van Renesse [96], increases the write availability for cases where one or more node failures, result in the data becoming unavailable for update, because a write quorum can no longer be acquired. The idea is to replace crashed nodes with processes called ghosts. Ghosts have the same number of votes as the failed node, but do not hold any data, they can be thought of as dynamically generated witnesses.

Although there is an increase in write availability over the witnesses method, implementation of this technique necessitates some rigid assumptions about the network and protocol architecture:

- Network can only partition at gateways or bridges that connect *segments.*
- Segments, which are collections of nodes, cannot be partitioned.
- If a segment is down (i.e. communication link has failed) nodes within that segment cannot

communicate with each other or with nodes on other segments.

These assumptions are demanding since networks can partition into subnetworks (segment partitions). Communication links may fail in one direction and not another, which creates a scenario where a node can send information but not receive it. If the segment is partitioned then some nodes may be able communicate with nodes in other segments but not with nodes within their own segment.

Failures are detected on each segment with a boot service which monitors the status of each node by polling them at regular intervals. This service must be replicated. However, Van Renesse [96] argues that, since the segments cannot be partitioned, the boot service can be controlled by either Voting or Available Copies techniques. If the boot service becomes unavailable, the algorithm reverts to Weighted Voting.

The read availability remains the same, since ghosts cannot participate in a read quorum.

3.6.3 DYNAMIC VOTING

Dynamic Voting proposes an extension to the original majority voting concept [61]. As well as the version number, each copy has an integer variable called Update Nodes Cardinality. This number represents the number of nodes that participated in the most recent update. An update is accepted if it can collect more than half of the up-to-date copies of the file. Once the update is accepted, the version number of the participating copies is incremented which has the effect of reassigning votes in such a way that nodes not participating in the update receive no votes. At the participating nodes (those with votes), the update nodes cardinality is the total number of votes currently available. A node with no votes can regain its vote when it rejoins a partition that allows it to participate in an update.

For example consider a network with five copies, each having one vote and assume that the file has been updated 6 times, the following table illustrates this

	A B C D E

Version Number	6	6	6	6	6
Nodes Cardinality	5	5	5	5	5

Now let us suppose that the system has partitioned into two sub-networks and a user has requested an update

	A	B	C	D	E
Version Number	6	6	7	7	7
Nodes Cardinality	5	5	3	3	3

Up to this point, the majority voting approach would have accepted the update in the second partition (the one containing the set of copies {C,D,E}). However if the system partitions again as follows

	A	B	C	D	E
Version Number	6	6	7	8	8
Nodes Cardinality	5	5	3	2	2

Majority voting would reject another update on the basis that there is no partition containing a majority of the votes. Using the dynamic voting update nodes cardinality another update is accepted since the partition containing the set of copies {D,E} holds a majority of current copies.

Dynamic-linear Voting, extends the Dynamic Voting algorithm by adding a third variable, the distinguished node. Where there are an even number of nodes participating in an update, each

node sets its distinguished node entry to name one of the participating nodes. To illustrate this, using our example, if we linearly order the nodes in this form A>B>C>D>E, and use this order to determine a distinguished node, the previous partition would have a distinguished node.

	A B	C	D E
Version Number	6 6	7	8 8
Nodes Cardinality	5 5	3	2 2
Distinguished node	- -	-	D D

Then if a further partition occurs that separates D and E, the system would change to the following:

	A B	C	D	E
Version Number	6 6	7	9	8
Nodes Cardinality	4 4	3	1	2
Distinguished node	- -	-	D	D

At this point not even dynamic voting would accept an update, but for Dynamic-linear Voting the partition containing the distinguished node D can process updates (since node D is ordered higher than node E).

This example shows that node and link failures can occur in such a way that Dynamic and Dynamic-linear schemes can accept updates that would be rejected by Majority Voting. The reverse can also be true. For example, suppose that node D fails and the remaining four nodes regroup into a single partition. Then updates are blocked under both dynamic and dynamic-linear Voting, while Majority Voting would accept an update in the partition containing the set of copies {A,B,C,E}. Jajodia and Mutchler [61] compare (using an analysis based on

stochastic processes) the dynamic voting techniques against the main static ones, and conclude that dynamic voting has better availability.

There are no discussions in the literature of the comparative reliability offered by these two types of replication techniques, which can be shown to present more significant differences than the variations in their availability.

The differences in availability comparisons are small. There is no discussion of the comparative reliability offered by these two systems which this thesis shows may often reveal more significant differences than tiny variations in availability.

3.7 TREE QUORUM

Agrawal and El Abbadi [1] have proposed the tree quorum concept. It combines the high read availability characteristics of the unanimous update method with the higher write availability advantage offered by the voting techniques.

The copies are arranged in levels according to a tree structure. The root being the first level, its children being the second level, etc.

Write Operation

A write operation must write to a majority of the copies at all levels of the tree.

Read Operation

A read operation can be executed by accessing a majority of copies at any single level of the tree.

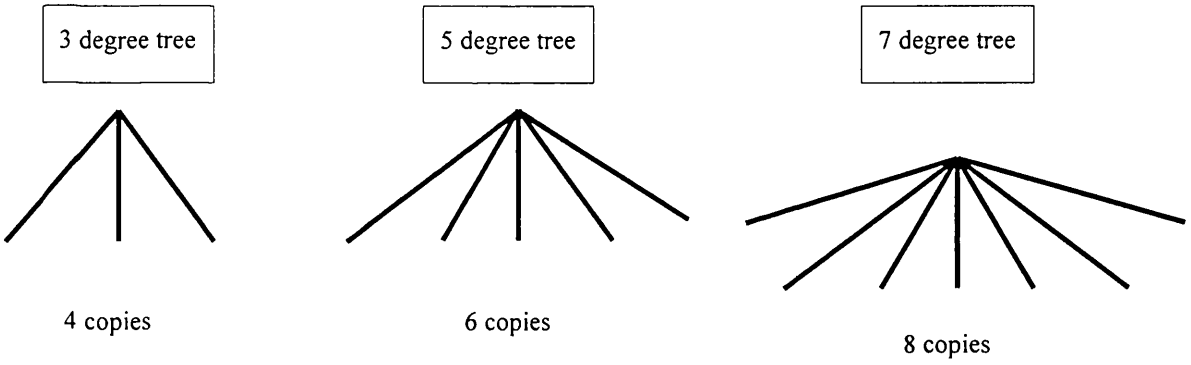
Hence, it is clear that a read operation has at least one copy in common with any write operation. This means that if the root is available, a read operation can execute by accessing only one copy (as with Unanimous Update), but a write operation does not need all the copies to be available; it can execute after the failure of several copies.

The idea is creative, but has two significant restrictions:

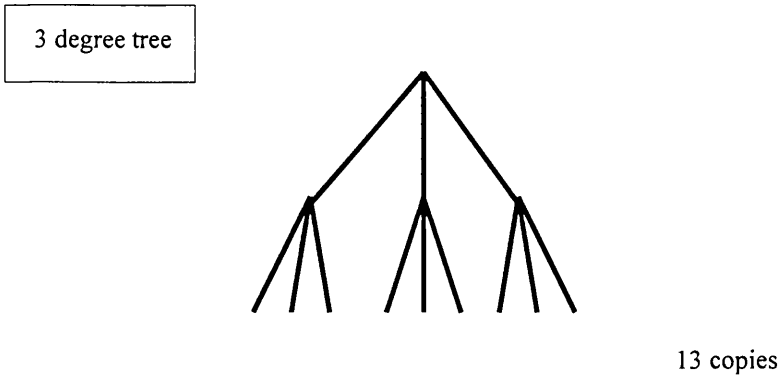
1. the degree of each node in the tree must be $2d+1$, for some integer $d > 0$,

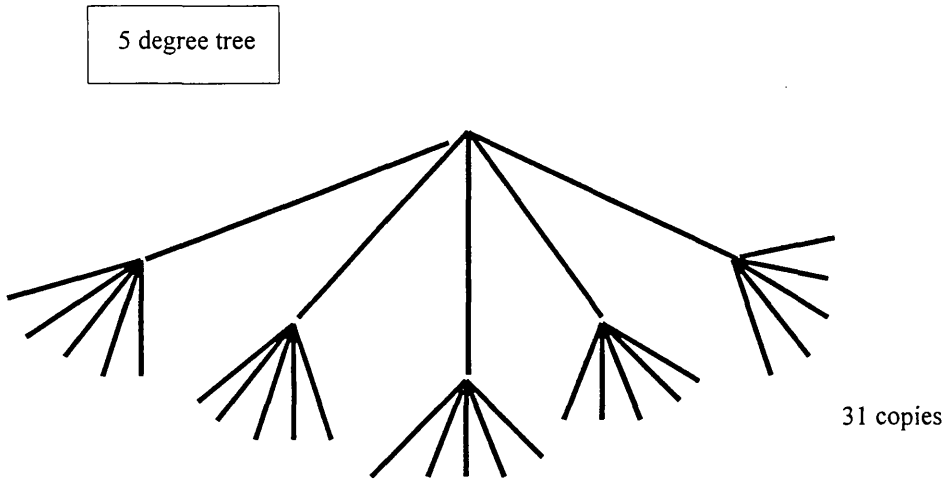
2. the tree must be complete (this means that it must have the maximum number of nodes).

Therefore the minimum number of copies before the protocol can be used at all must be 4. If more copies are needed they cannot be chosen according to needs alone, they must fit the tree structure restrictions. For example, if only two levels are used then one could have any even number of copies 4 or more.



However, in order to have higher write availability it is advantageous to have more levels.





A three level tree could have any number of copies in the set {13,31,57,...,} (i.e. $(2d+1)^2+(2d+1)+1$, for integer $d>0$), which could be costly for large data objects due to storage costs.

3.8 REGENERATION

Pu [91] first introduced the idea of regenerating replicas to replace those lost due to node failures as a technique for increasing the availability of replicated data objects in the Eden System [81]. He proposed an algorithm that provides serial consistency in a partition-free system as an extension to the Available Copies algorithm. It creates new replicas to replace those lost due to system failure when it detects that one or more of the replicas have become inaccessible.

The notion of regeneration is simple and efficient. However in Pu's proposal it carried with it the same weaknesses associated with the Available Copies algorithm. Read transactions are allowed to continue as long as one current replica of the object remains accessible. If fewer than the initial copies are accessible for a write transaction, then new copies are regenerated

on other available machines. If there are no spare machines, then the write transaction is rejected. In the case of total failure, manual intervention is required and consistency is not guaranteed under partitioning.

The idea of regeneration has become a familiar addition to many replication methods, since it is possible to combine it with most algorithms to increase the availability. Long and Carrol [76] have done an interesting study of the reliability of regeneration applied to available copies, static voting, and dynamic voting. They have concluded that with five or more participating nodes and a high rate of regeneration, replacing one replica with a regenerated copy was shown to have a slight detrimental effect on the reliability. However, the reliability reduces drastically if fewer than five copies are maintained.

The costs associated with regeneration based replication control techniques include a significant increase in network message traffic. Regeneration follows every node or partitioning failure. The cost of transmitting the current copy in order to regenerate one on another node may be excessive for large data objects. Long and Carrol state that regeneration may be best suited for small data objects with strong fault tolerance requirements.

3.9 DISCUSSION

Many of the Consistency Control Techniques outlined in this Chapter were considered for the purposes of this work. After careful deliberation, three of the techniques were chosen, namely, Weighted Voting (WV), Voting With Witnesses (VWW) and Voting With Regeneration (VWR).

WV was chosen as the standard with which to gauge VWW and VWR. VWW was chosen as the reduction in storage cost is extremely attractive when choosing a voting replication technique, thus it is important to understand the reliability of WV compared with VWW. Regeneration offers an increase in availability when applied to any replication procedure, and a measure of the effect of regeneration on the reliability of VW will be an important step in

understanding the comparative reliabilities of other regenerative techniques. All three algorithms offer consistency in the face of network partitions and are popular in the literature.

Chapter 4

Measures of Comparative Reliability and The Presentation of experimental data

This chapter describes a selection of pessimistic replication control techniques, measures of comparative reliability and a new technique for data presentation.

Reliability is not an easy measure to manipulate. It is not easy to compare reliabilities because the reliability of an object (whether replicated or not) is a function of time intervals.

Assuming that network links can fail, the reliability of an object is then dependent on the subnet from which the object is accessed, and in the case of a replicated object, on the placement of copies (or witnesses) and on the configuration (or topology) of the network. In order to characterize fully a particular configuration, the failure and recovery properties of individual nodes (at least of those that hold copies) and of each link between subnets needs to considered. It is obvious that there are many components or parameters to consider in analysing the reliability. Furthermore, in analyzing and comparing the reliabilities of objects one must be able to exhibit variations in the reliability in such a way that simplifies the number of variables effectively without neglecting any significant outcome.

This chapter presents solutions to these problems by introducing the simple measure of comparative reliability and a graphical notation that permits all the numerous components to be presented simply in a single diagram.

4.1 RELATED LITERATURE

There have not been many investigations in the literature that have included reliability as a measure of a system's performance.

Long and Carroll [75] investigated the reliability of regeneration based replica control protocols both analytically and using simulation. As part of their investigation they examined the reliability of various voting techniques for system configurations in which partitioning failures can occur. They used different ratios of replicas and spares to show that Dynamic Voting (DV) is more reliable than Majority Voting (MV). However, they did not consider the effect of the topology on the location of copies and how the different consistency controls would react to replacement or transfer of these copies.

Bek [12,13] compared the availability of MV and the Reliable Histories (RH) replication techniques using topologies that consisted of two subnets linked together. She considered the variance of availability as a function of copy placement and showed there was variation in the behavior of these algorithms to copy placement.

Bek went on to compare the reliability of a non-replicated file with one replicated using Available Copies (AC), Reliable Histories (RH), MV and the non replicated case, in a system that does not partition. The results showed that the reliability improved for all the replication algorithms in varying degrees. The effect of partitioning on the reliability is considered using two different network configurations. Bek concludes that the reliability of all the algorithms was greatly reduced when the system allowed partitioning. She also considered the effect of copy placement, for two network configurations.

To summarize, Bek's studies indicate that:

1. the network topology (and by implication, the location of the client) has a significant effect on a replicated file's reliability,

2. the network topology has a significant effect on the choice for the location of copies,

3. reliability demonstrates differences between algorithms and configurations much more dramatically than the tiny variations in availability that are reported in the literature.

There are indications therefore, that the replication algorithms' performance may vary when copy placements are altered in any particular network configuration.

4.2 COMPARING THE RELIABILITY OF DIFFERENT SYSTEMS

In Chapter 2 a distributed system was defined as a set of subnets that contain a fully connected set of nodes joined together by links. In examining the reliability of one system compared with another there are many aspects to consider. One of the most significant is the fact that the panorama or view from one subnet may be different to that of another subnet. For example consider subnets S_1,S_2,S_3,S_4 connected together in a chain by links l_1,l_2,l_3 (Figure 4.1).

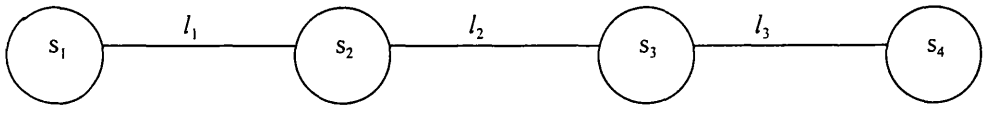


Figure 4. 1 Four subnets in a chain

If S_1 is partitioned, i.e., l_1 is down, then the view from S_1 will be only of its own nodes, and the view from S_2, S_3 and S_4 will be of each other's nodes but not of S_1's.

Consider a data file placed on one of the nodes in S_1. In order to access the file from S_2, l_1 must be up. In order to access the file from S_3, l_1 and l_2 must both be up. Consequently, the reliability will be higher when viewed from S_2. Figure 4.2 shows the reliability functions as viewed from S_2,S_3,S_4, with probability of the link being up of 0.8.

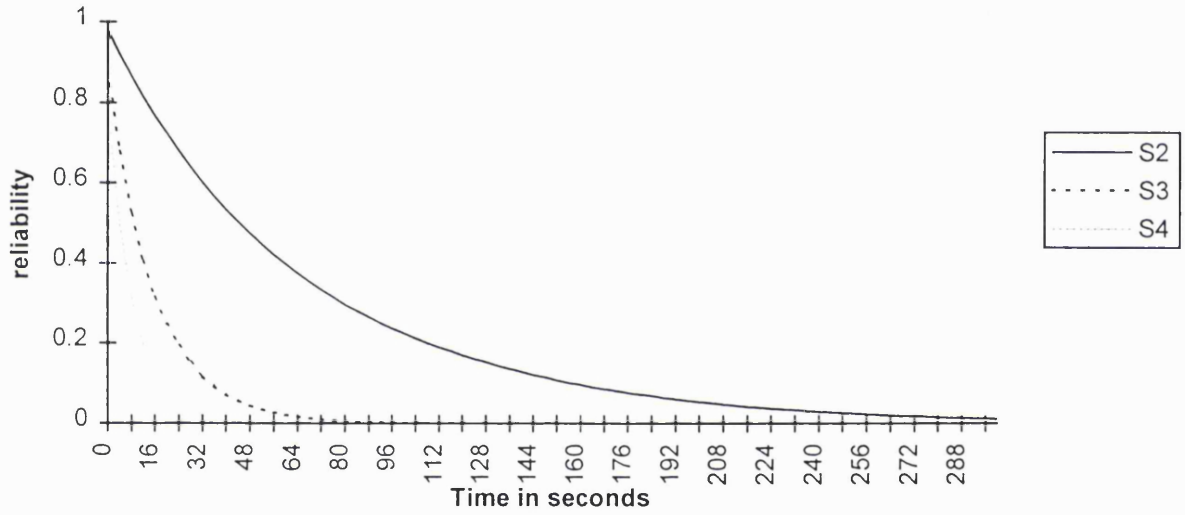


Figure 4. 2 Reliability Functions of 4 subnets connected in a chain with probability of a link being up of 0.8

It is easy to see that reliability from S_2 is much higher than that from S_3 or S_4 and reliability from S_3 is higher than that from S_4. We can also say that the reduction in reliability is greater between S_2 and S_3 than between S_3 and S_4.

Fig. 4.3 and Table 4.1 illustrate an example of higher link availabilities. Using $R(\delta t)=Ae^{-\lambda\delta t}$ to calculate the reliability with MTTF=99.5, MTTR=0.5, λ=(1/99.5)=0.01005025 for the chain topology in Fig. 4.1.

δt	R(δt) : (S2)	$(R(\delta t))^2$:S3	$(R(\delta t))^3$:S4
0	0.995	0.990025	0.985075
20	0.81381878	0.662301	0.538993
40	0.66562915	0.443062	0.294915
60	0.54442362	0.296397	0.161366
80	0.44528861	0.198282	0.088293
100	0.36420526	0.132645	0.04831
120	0.29788651	0.088736	0.026433
140	0.24364386	0.059362	0.014463
160	0.19927834	0.039712	0.007914
180	0.16299141	0.026566	0.00433
200	0.13331203	0.017772	0.002369
220	0.10903702	0.011889	0.001296
240	0.08918229	0.007953	0.000709
260	0.07294293	0.005321	0.000388
280	0.05966063	0.003559	0.000212
300	0.04879693	0.002381	0.000116

Table 4.1 Reliability calculations for subnets in a chain.

Note that the availability, R(0), is 0.995.

The observations above are repeated exactly for higher availability, in Fig 4.3. Again the reliability from S_2 is much higher than that from S_3 or S_4 and reliability from S_3 is higher than that from S_4. We can also say that the reduction in reliability is greater between S_2 and S_3 than between S_3 and S_4.

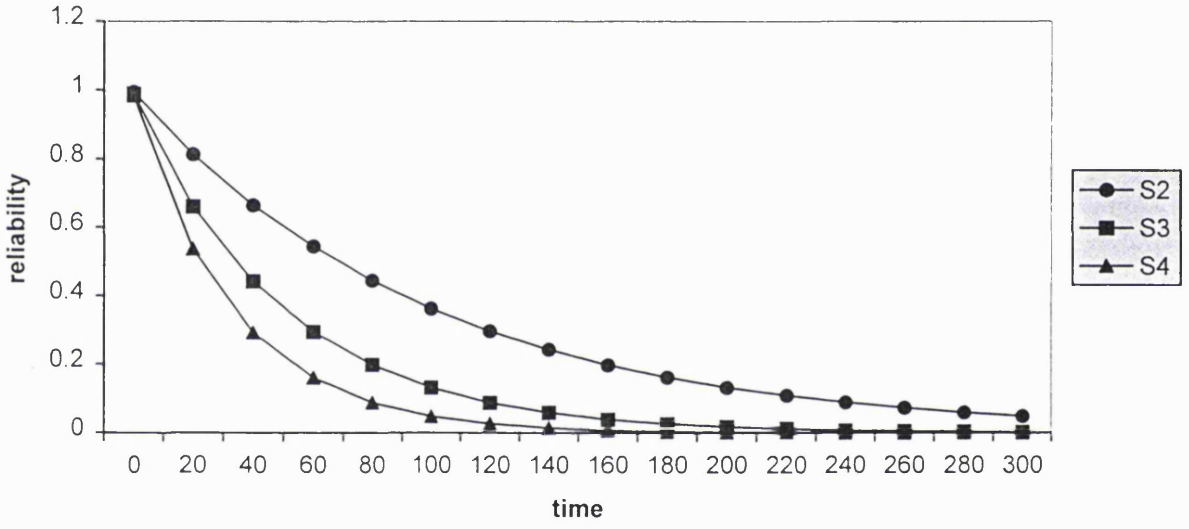


Fig 4.3 Reliability for chain calculated using formula with availability 0.995.

Such observations are important. However, if one wanted to compare the reliability of another system configuration with this configuration using the same form of representation, the picture would have to include all the reliability functions pertaining to the different subnets in the new system. This will be both confusing and tedious when a large number of subnet views are involved.

Comparing the reliability functions in this manner is therefore ineffective. A more useful measure of reliability would be one that contains all the significant information of the reliability function in one value. Even if such a value were to have no intuitive interpretation, it could be used to study the relationship of one subnet view relative to another or indeed to compare any two reliabilities.

4.3 MEASURES OF COMPARATIVE RELIABILITY

Inevitably, the reliability function for some object over a period of time must have some form of decay. In fact, the reliability functions of distributed systems, in example fig 4.2, follow a similar pattern. The availability, *R(0)*, is usually close to 1. There is then a decrease that may be fast or slow followed by a tail that eventually tends to 0.

Gnedenko [40] established, that one could assume that the failure rate of a machine, λ, is constant. He shows that if λ is constant, then the reliability for this machine is (asymptotically) negatively exponential. Therefore reliability can be approximated by a function of the form $Ae^{-\lambda\delta t}$, at least for large δt.

In finding an acceptable comparative reliability measure it is imperative to retain as much of the information of the reliability function as possible. If any information is lost as a consequence of the summarization process to simplify the presentation and analysis of reliability information, one has to be sure that this is clarified and that this "lost" information does not contribute in a significant way to the results (with respect to the objectives). It is also important that any process that transforms the reliability function must be applicable, practical and if possible, intuitive. This is because the reliability of most systems can not be derived analytically. Data to represent the reliability can be collected in experimentation either from real systems or using simulation. The data can be summarized in a graph for each subnet, then these graphs can be summarized further into a single number, which, can be termed the *Relative Reliability*. These relative reliability numbers can be listed in a table against their subnet reference and compared to each other. Reducing the reliability to a single number allows more complex data to be presented in graphs

without requiring a third axis (or dimension). For example, the relative reliability against the repair/failure rate.

4.3.1 DECAY FACTOR

Consider a reliability function, which can be approximated by $Ae^{-\lambda\delta}$. The *decay factor*, λ, can be used on its own as a measure of reliability. For example consider the scenario above of the network configuration in a chain, with subnets S_1, S_2, S_3, and S_4. We can use the decay factor for each of the reliability functions and plot a graph of the decay rate against link availability as shown in Fig 4.4

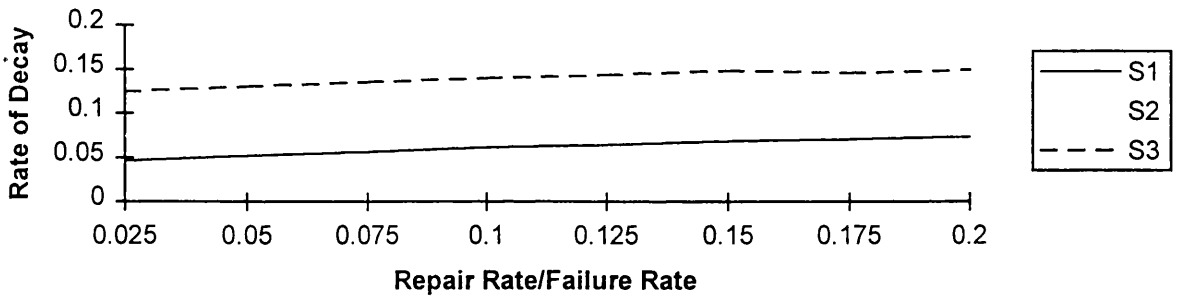


Figure 4. 4 Using the decay factor - 4 subnets in a chain

Using a comparison of the gradients of each reliability function highlights the differences in the rate of decay as viewed from each of the subnets. Already more information has been included in

the figure (Fig 4.4), the reliability as viewed from each of the three subnet decays similarly and the decay is greater for lower availabilities.

In the case of a topology that consists of a chain of subnets, it is easy to see that the reliability fits the assumption of a negative exponential function. However, in more complex topological configurations the assumption that the reliability is negative exponential may not be correct. There are situations where this approximation may be too stringent since reliability curves are "flattened" near the y-axis and vital information may be ignored or lost. The reliability function was estimated by linear regression using the log of δt. In more complex situations, if the graphs do not match the negative exponential function, some arbitrary form of curve fitting may be required. In these cases, this measure becomes more arbitrary and less intuitive. After some careful considerations, this measure was abandoned.

4.3.2 MAXIMUM DIFFERENCE

The motivation for considering the maximum difference as a measure is the fact that it accentuates the differences between systems with very similar reliabilities. One way to do this is to compare two reliability functions by calculating their ratio at the point of maximum difference.

Assuming that the reliability function can be approximated by an exponential function (as above), and given any two reliability functions $R_1(t)=Ae^{-at}$ and $R_2(t)=Be^{-bt}$, their difference, R_1-R_2, has a maximum at

$$t_{max} = \frac{\ln(Aa/Bb)}{a-b}$$

Therefore the ratio

$$\frac{R_1(t_{max})}{R_2(t_{max})}$$

gives a measure of how much better (or worse) R_1 is than R_2 in the worst case. Using the maximum allows otherwise small differences to be made as dramatic as possible, furthermore the value t_{max} itself may be of significance.

Problems occur when several reliability functions are compared. If R_1 and R_2 are at their maximum difference at time interval T_{12}, and R_2 and R_3 are at their maximum difference at time interval T_{23} , it may lead to misleading interpretation of the reliabilities if the maximum differences are used and T_{12} is very much different to T_{23}.

With experimental data, the functions will have to be approximated by some method such as interpolations. When the maximum difference occurs at a point where the approximation was more vulnerable to imprecision, comparing this maximum may also lead to misleading interpretations about how much more reliable one view is from another.

Fig 4.5 uses this measure to describe the same system as Fig 4.1.

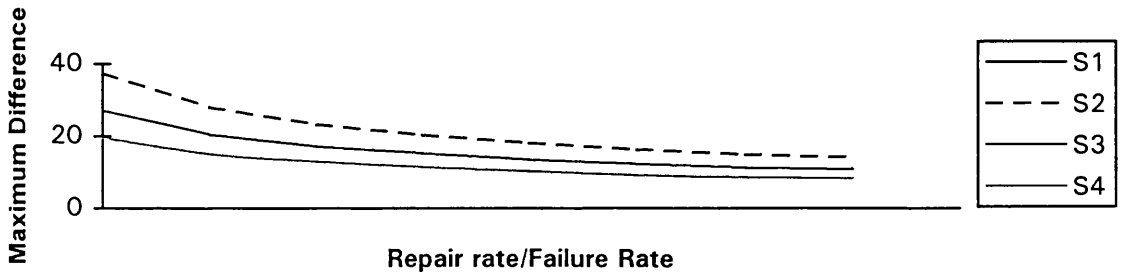


Figure 4. 5 Maximum difference: 4 subnet in a chain

It is easy to see that the higher availabilities give rise to higher comparative reliabilities.

4.3.3 AREA RATIOS

Maintaining the supposition that less reliability information will be lost by considering the function as a whole, another measure can be developed by manipulating the area under the reliability function.

Assuming the approximation of a reliability curve by an exponential function of the form Ae^{-at}, then the area under the curve is

$$\int_0^\infty Ae^{-at} = \frac{A}{a}$$

Consider two reliability curves with areas A/a and B/b. Then the ratio of the areas between functions is Ab/aB.

The actual value of the area of a reliability function on its own may not be very interesting. However, by taking the ratio of one area to another, we can compare how much more reliable one subnet view is than another. If R_1 is x times more reliable than R_2, and R_2 is y times more reliable than R_3, R_1 is xy times more reliable than R_3.

For example consider the network configuration in Figure 4.1 of the topology for a chain with subnets S1, S2, S3, and S4. Suppose we place a single replica of an object in subnet S1. We assume the probability of any link (l_1,l_2,l_3) being up is 0.8 and the Mean Time To Failure is 20 time units (which may be hours, days or whatever is appropriate for the application). It is obvious that the availability from subnet S1 is 0.8, which is equivalent to the reliability form S1 for 0 time. Using the formulae above the reliability from S2 is

$$0.8e^{-0.05t}$$

the reliability from S3 is therefore

$$(0.8)^2e^{-2(0.05)t} = 0.64e^{-0.1t}$$

while the reliability from S4 is

$$(0.8)^3e^{-3(0.05)t} = 0.512e^{-0.15t}$$

Now the area under the curve for S2 = 0.8/0.05 = 16.0

S3 = 0.64/0.1 = 6.4

S4 = 0.512/0.15 = 3.413

Using the areas to compare reliabilities, S2 is 2.5 times more reliable than S3 and S3 is 1.88 times more reliable than S4. Therefore S2 should be 4.69 more reliable than S4, and checking the figures above it is exactly that.

It is important to understand that this relative reliability measure must be used with a reference reliability. The integral of this reliability is used in calculating the relative reliability of other objects or views. For example, taking the reliability of a copy (unreplicated file) as the reference, the relative reliability of an object as seen from different points in the simple chain topology (Fig 4.1) can be plotted against the link availability.

This measure can offer a splendid strategy for understanding the comparative reliabilities for different network topologies and placement of copies. It can also offer a great opportunity to study partitioned networks, since we can compare the reliability of network topologies that can partition against the reliability of networks that do not.

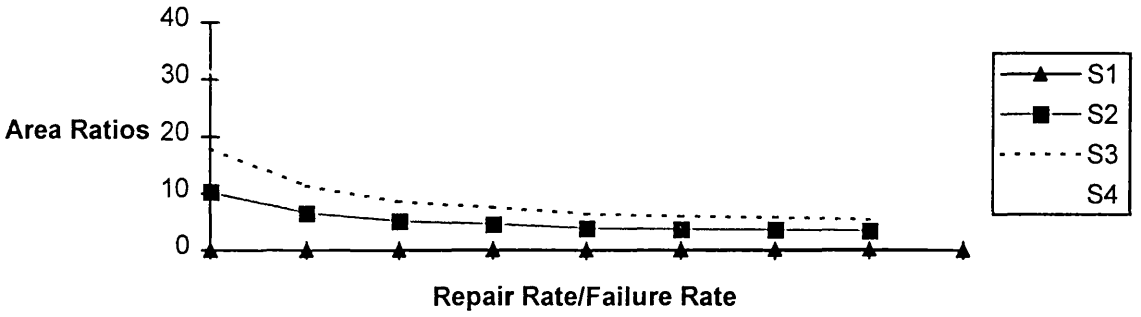


Figure 4. 6 Area Ratios: 4 subnets in a chain as a function of the Repair/Failure rate

Using the same scenario as was employed for the Decay factor in Fig 4.4 and for the Maximum Difference in Fig 4.5, Fig 4.6 illustrates the Area Ratios as a function of the link availability. The areas are plotted with reference to the area of the reliability function of one copy. For instance, when the link availability is 0.2 the values of the relative reliabilities are 12.8, 5.5, 3.5. One could deduce that the reliability from S2 is approximately 2 times better than that from S3.

As in the Maximum Difference example, it is easy to see that the higher availabilities produce comparatively higher reliabilities. It is simple to rearrange the representation using the availabilities (i.e. the probability of any node/links being up) Fig 4.7.

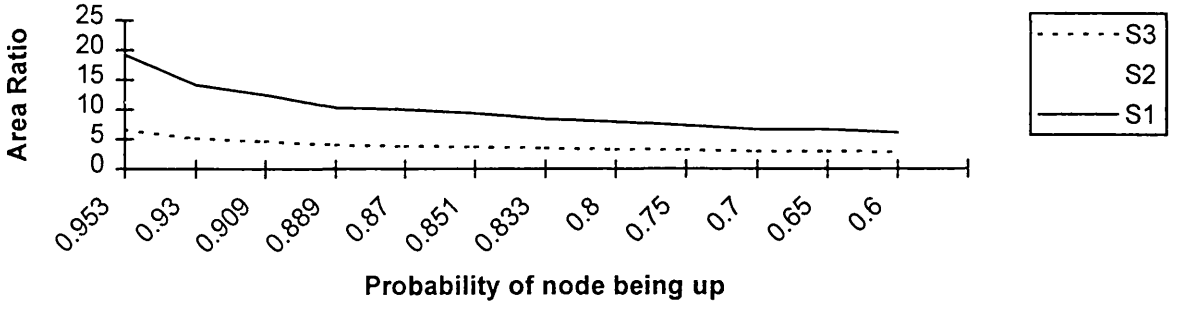


Figure 4. 7 Area Ratios: 4 subnets in a chain as a function of probability of node being up

Again the area ratios are with reference to the reliability of one copy. Notice that the values of the relative reliabilities are S2=8, S3=5, S4=3 for an availability of 0.8 which was calculated using repair/failure rates of 0.2.

The exponential approximation assumption can be relaxed by using a numerical method to calculate the area under a reliability function obtained by simulation of observation.

Chapter 6 includes a discussion on how effective this reliability measure is in factoring out variables that complicate the investigation. Two different techniques of collecting reliability data from equivalent network configurations will be used to generate results. Using the area ratios with a corresponding reference area will generate identical results for both techniques.

Area ratios were chosen as the measure for this investigation. It will be used extensively in conjunction with a novel presentation technique described below.

4.4 NOTATION

In the previous section graphs are used to illustrate the view of reliability from different subnets for various availabilities. There is an abundance of information in these graphs already, but it is not enough. It is not easy to see that network configuration or topology was used. The topology of four subnets in a chain had to be described separately. Once one delves more deeply into the investigation of systems, it would be necessary to show the placement of copies within the topology. One would need to be aware of whether the links fail or whether they are perfect (i.e. always working) for investigations that do not allow partitioning. In the case of voting with regeneration one would need to investigate if altering the position of the supplementary servers affects the reliability. The position of these servers would have to be included in any representation. A presentation technique that shows the relative reliabilities of the replicated service as they would be perceived from a client on each subnet. A technique that would clearly display the topology of the network and the placement of copies within that is essential for this investigation.

After several attempts at representing the data the following presentation technique was developed. A diagram showing the network topologies as well as other information represented by symbols will represent the data collected. The following table defines the symbols used in presenting results:

- The view from a subnet that has infinite reliability is represented by a hollow square. These subnets have nodes that do not fail and contain all the copies.
- Subnets that contain nodes that do not fail, are represented by hollow circles with a full line.
- Subnets that contain nodes that fail and recover, are represented by dotted hollow circles.

- The hollow circles (denoting subnets) will vary in size according to the area that is attributed to the view of the comparative reliability from this subnet with respect to some reference area. For instance if the comparative reliability as viewed from one subnet is twice that of the reliability as view from a second subnet, the two subnets will be represented by two circles where the area of one is twice the area of the other.

- Subnets are connected by links represented by lines, where broken lines indicate that a link may fail and repair.

- Small circles, with the center filled in represent the copies or replicas. They will be placed inside a subnet to show their position within the network topology.

- Small circles, that are hollow, represent witnesses, in the case of Voting With Witnesses.

- Any supplementary nodes (they do not hold copies, but may be used for the regeneration of copies) will be represented by filled in triangles. They will be linked by a line to the subnet that they belong, to show their position in the topology.

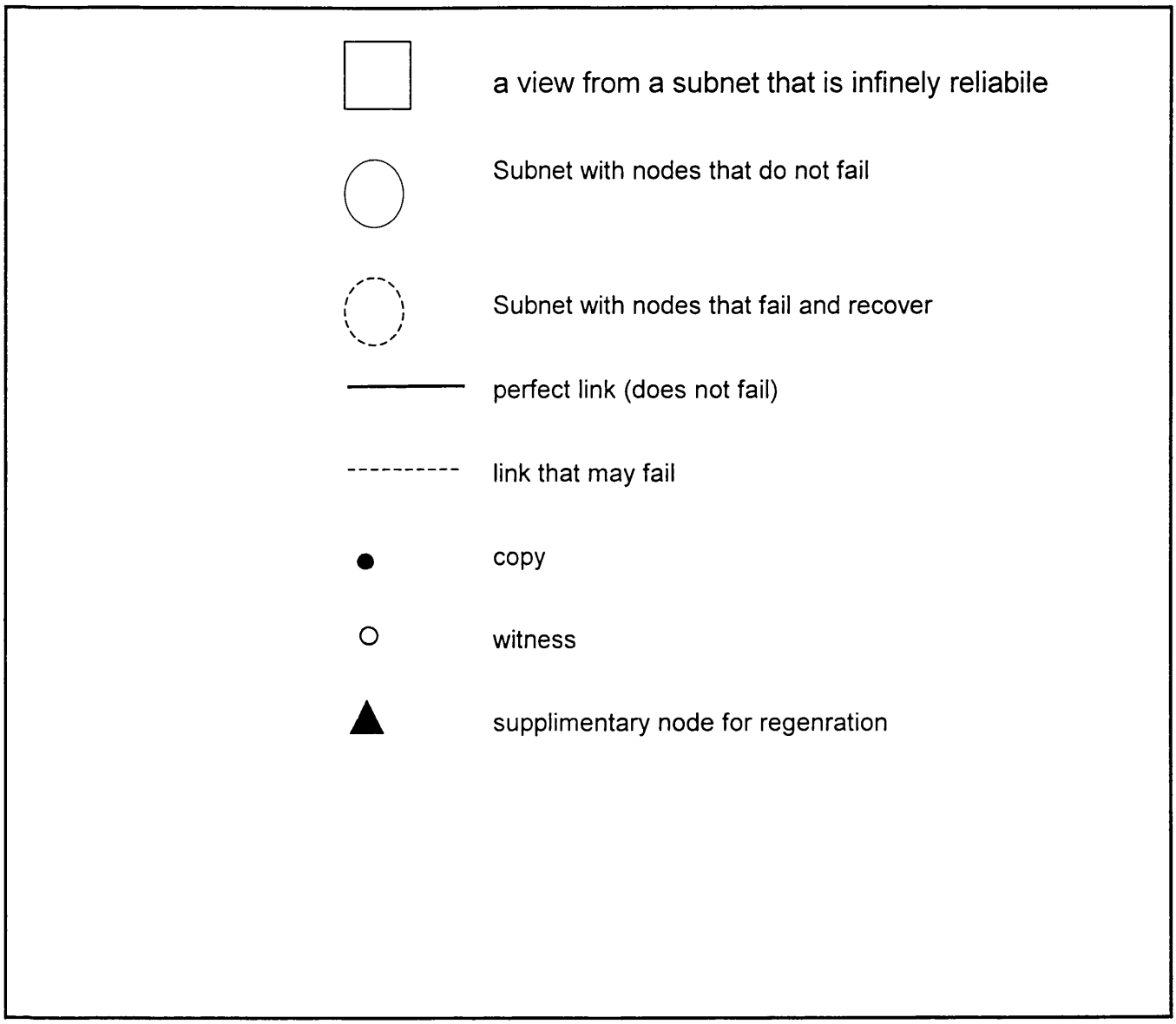


Figure 4. 8 Key to notation technique

Using this technique the scenario that has been described above employing the Area Ratios measure, can be illustrated as in Fig 4.9.

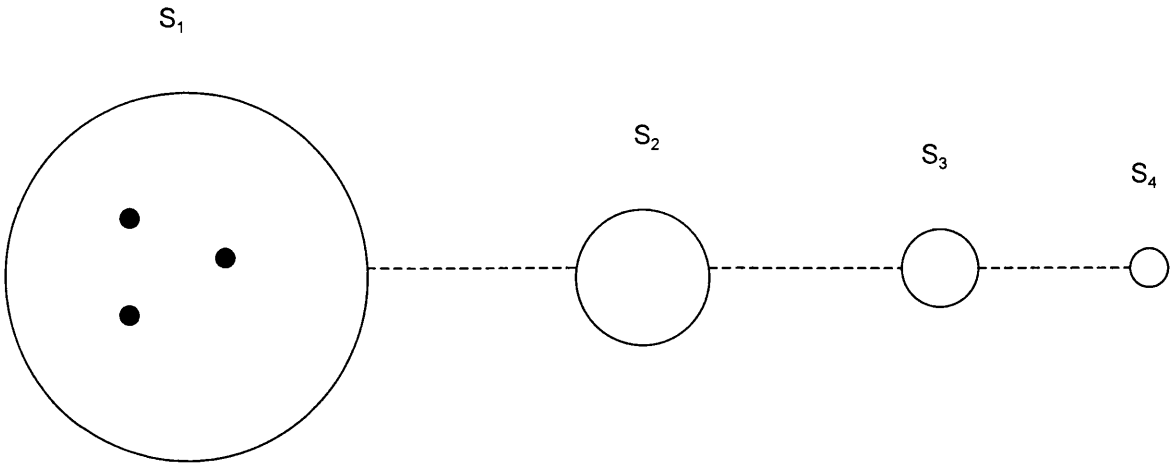


Figure 4. 9 An example of the notation technique using the Area Ratios

At a glance we are able to discern:

- That the topology is a chain linking four subnets,
- that each subnet is made up of nodes that do not fail (without loss of generality, by definition the links connecting the nodes in each subnet do not fail)
- that the links connecting the subnet may fail and recover,
- that there are three replicas placed in the first subnet in the chain, S_1,
- that there are no other replicas in any of the other subnets
- that the reliability is greater from subnet S_1, compared with the reliability of the other subnets
- that the reliability decreases as one gets further from the subnet with the copies.

The diagrams do not contain all the information required to understand the experiment. Where the distribution of failure and repair times of both copies and network links, as well as the algorithm used to control access to copies cannot be determined from the context, they will be provided in the text.

4.5 SUMMARY

This chapter describes the limitation of conventional methods of representing reliability. It describes measures of comparative reliability that have been considered for condensing the reliability function. Examples of each measure are and given. The author's choice of the Area Ratios measure is used with a accommodating notation technique that enables the simultaneous representation of numerous parameters that may be used in the investigation of the resiliency of replication techniques.

Chapter 5

Approaches to the Collection of Reliability Information

For the purposes of this work data can be collected by mathematical modelling, or by experimentation, where the latter can be performed on either real or simulated systems. This chapter discusses these approaches and explains the author's decision to use simulation.

5.1 WHY SIMULATION?

Simulation is a tool that has become very popular in recent years. As an investigation technique it is versatile and can be used in many different applications from simulating world economic conditions and disease control strategies to space system reliability. It can be described as a technique that ``takes on the characteristics of reality".

Figure 5.1 displays a set of three alternative approaches that can be used in the designand analysis of systems. At the left extreme of the figure is experimentation on the real system, and at the right extreme is analytical or mathematical modelling.

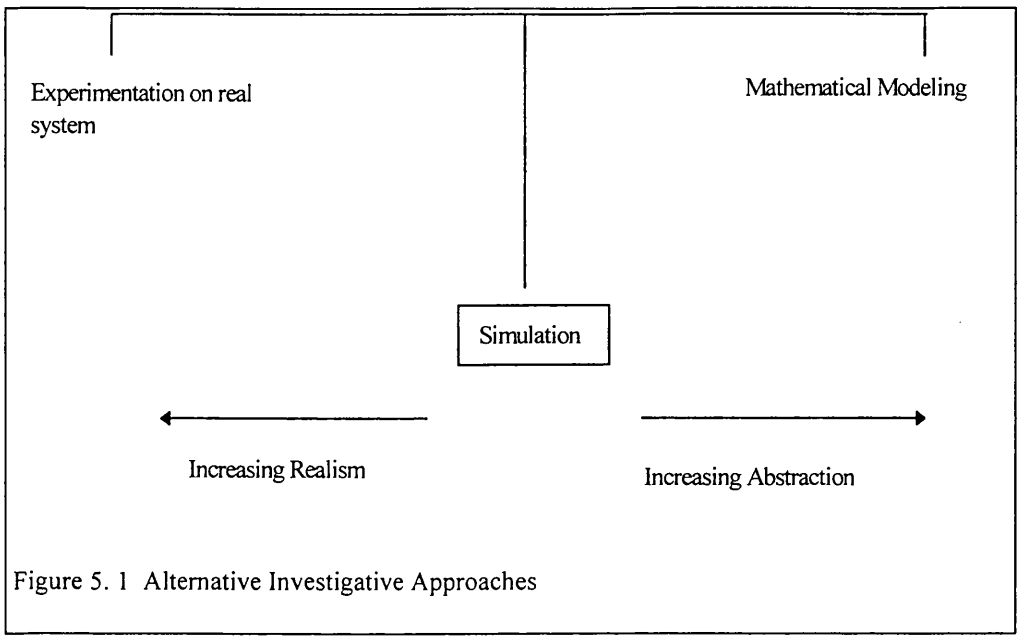


Figure 5. 1 Alternative Investigative Approaches

5.1.1 EXPERIMENTATION ON A REAL SYSTEM

On the surface this approach is realistic and it would seem to lead to the most significant results were it possible to implement. However, there are several reasons why it could not be adopted.

- It is extremely difficult to perform controlled experiments on a real system. There are elements beyond our control, which means the results produced by a real system could not be relied upon to give exact answers, but only estimates.

- Performance experiments are not only costly, they may also interfere with the day to day running of the system. This would limit the amount of experiments that one could run and reduce the scope of the investigation.

- It would be extremely difficult to isolate particular aspects of a system for closer investigation, or for simplification.

- If a system could be found where time and money were not limiting factors, then investigators would be limited by the existence of this particular system. They would therefore, not be able

to investigate any concepts that were not compatible with the implementation already in existence in this particular system.

- Ensuring that the perceived view of a system at any one moment was correct would be very difficult, if not impossible, especially in determining whether the system is partitioned or not.

- There is no generality in the results, since inferences, deductions or conclusions made about one system are particular to that system. The results apply only to the system in the state in which the experimentation was performed, as there may be factors particular to that system that prejudice, influence or even distort the results.

- Incorporating, implementing and validating several different replication algorithms into a particular system may be extremely costly, time-consuming, and hazardous.

- Real system experiments would be very slow since failures are rare. If the failure rate were therefore artificially increased, the system would loose its realistic nature.

5.1.2 MATHEMATICAL MODELING

Mathematical modelling can be divided into 'analytical' and `numerical' methods. Mitriani [80] defines an analytical solution as "providing a closed-form expression for the desired system characteristics in terms of the defining parameters". Mitriani states that such solutions are usually unobtainable for any but the simplest of models. Although in principle numerical solutions can be applied to models of arbitrary complexity, they do have the disadvantage of providing results only for individual cases. It is difficult to draw generalised conclusions and form a deeper understanding of a generalised behaviour when one is using individual results.

It is moderately easy to find analytical results for the availability of even quite complex systems. It is difficult to derive analytical solutions for the reliability of even very simple systems. However, even if it was possible to approach the study of the resiliency of systems analytically, the number of parameters one is able to consider would be restricted, limited by the methodology of the particular analytical process being used. This approach can hinder any creative thinking where radical changes (if only experimental) are often useful.

5.1.3 ANALYTICAL AVAILABILITY

Generally, in order to derive availability one would use an analytical or mathematical model. There are methods that are suitable for particular algorithmic structures such as K-out-of-n reliability theory [11,21,25,44,56,65,68] where a system is operative if and only if at least *k* of its *n* components are operative. There are more general methods for calculating the availability between any given pair of nodes in a network topology. SYREL [48] is such a method, it is based on path and cutset methods of a graph representing the network topology.

In a network several paths may exist between a given pair of nodes, which are denoted by P_i' s. It is assumed that the failures of nodes are statistically independent and p_k is the probability of a node being up. The availability between a pair of nodes is given by

$$A = P(\overset{m}{\underset{j=1}{Y}} E_j)$$

where E_i denotes the event in which path P_i is up and *m* represents the number of paths between the two nodes. The availability can be calculated using a method that is based on decomposing the set of paths into another set of mutually exclusive paths. For example, consider the network in fig 5.2.

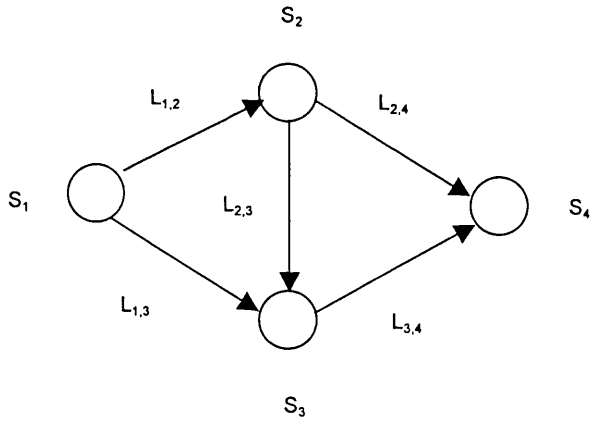


Figure 5. 2 A network with three paths

It consists of three paths $P_1 = I_{1,2} \cdot I_{2,4}$, $P_2 = I_{1,3} \cdot I_{3,4}$ and $P_3 = I_{1,2} \cdot I_{2,3} \cdot I_{3,4}$. The availability consists of three terms, which correspond to three mutually exclusive events:

- P_1 is up
- P_2 is up and P_1 is down
- P_3 is up and both P_1 and P_2 are down.

Using the equation above the availability, A, of this network is the sum of the terms corresponding to these three events

$$A = p_{1,2} \cdot p_{2,4} + p_{1,3} \cdot p_{3,4} \cdot (1 - p_{1,2} \cdot p_{2,4}) + p_{1,2} \cdot p_{2,3} \cdot p_{3,4} \cdot (1 - p_{2,4}) \cdot (1 - p_{1,3})$$

5.1.4 ANALYTICAL RELIABILITY

The analytical derivation of reliability is a much more difficult proposition than that of availability. The simplest case to analyse is that of a network connected in series. Let us simplify this even further by considering a system where the nodes do not fail, but the links fail and recover according to some failure and recovery distributions.

For this simple case let a computer network be represented by a graph *G(S,L)*, where *S* and *L* are the set of nodes and edges that represent the machines or subnets and the communication links, respectively. Consider the network in Fig 5.3 where the nodes $\{s_1, s_2, \ldots s_{n-1}\}$ are connected by links in a chain configuration $\{\lambda_1, \lambda_2, \ldots \lambda_n\}$ and all links fail independently of each other.

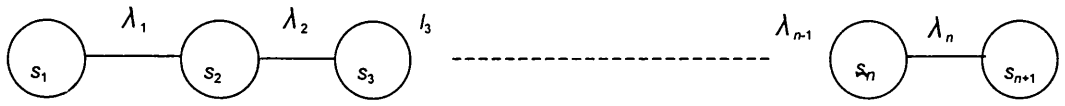


Figure 5. 3 Network System in a chain configuration

Since the *n* links in the network connect the machines in the form of a chain, the failure of an arbitrary link causes failure of the entire network.

For failure-free operation of the network for a period of time t, it is necessary that every link must be operational without failure during that period. Since the links are statistically independent we can calculate their reliability for period of time t as follows:

$$R(t) = R_1(t)R_2(t)...R_n(t)$$

for mutually exclusive events, where $R_k(t)$ is the reliability of the k^{th} link.

In particular, consider the case where each λ_k has failure distribution $e^{-\lambda t}$ and recovery distribution $e^{-\mu t}$ where λ is the failure rate and μ is the recovery rate. The reliability function of each link is of the form [40].

$$R_k(t) = Ae^{-\lambda t}$$

The reliability of this network connected in a chain is therefore

$$R(t) = A^n e^{-n\lambda t}$$

5.1.5 RELIABILITY OF A PAIR OF NODES CONNECTED BY A DOUBLE LINK

After looking at a network connected in series, the next step would be a system that includes some form of duplication or ring. The simplest topology of this type would be two nodes connected together by two identical links (Fig 5.4).

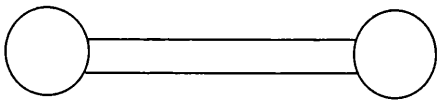


Figure 5. 4 Pair of nodes connected by double link

- This is proved in Chapter 2 by Gnedenko [40].

When one link fails messages can be sent by the other link. The only time the system is unavailable would be if both links were down simultaneously.

Let λ denote the failure rate of each link and let μ denote the repair rate. Let *R(t)* denote the probability of failure-free operation of a pair up to the instant *t*. The event that the double link system will operate without failure during the interval *(0,t)* can be decomposed into the following disjointed events [40]:

1. The first failure occurs after the instant *t*. The probability of this event is $e^{-2\lambda t}$

2. The first failure occurs prior to the instant *t*, then the second link takes over and fails after the instant *t*, but the first link has not recovered yet. The probability of this event is

$$\int_0^t 2\lambda e^{-2\lambda x}[1-e^{-\mu(t-x)}]e^{-\lambda(t-x)}dx$$

3. The first link fails, and the second link takes over, and when it fails the first link takes over again and the pair operate without failure for the remainder of the time until the instant *t*. The probability of such an event is

$$\int_0^t r(t-x)dx\int_0^x 2\lambda e^{-2\lambda z-\lambda(x-z)}[-\mu e^{-\mu(x-z)}]dz$$

One is able to obtain the desired probability by adding these events

$$r(t)=e^{-2\lambda t}-2\lambda e^{-\lambda t}\int_0^t e^{-\lambda x}[1-e^{-\mu(t-x)}dx-\int_0^t r(t-x)2\lambda e^{-\lambda x}dx\int_0^x e^{-\lambda z}[-\mu e^{-\mu(x-z)}]dz$$

If the time *t* is small in comparison with the Mean Time To Failure (MTTF) and Mean Time To Repair (MTTR), this complicated equation could be used to calculate the reliability since it converges rapidly. However, in order to study a system for a long period of time relative to the MTTF and MTTR, this equation is not practical at all. Gnedenko gives the following formula, which can be used in practice to approximate the reliability.

If the failure distribution is *F(t)* and the recovery distribution is *G(t)* then the reliability for a period of length *t* of a system can be approximated by

$$R(t) \approx e^{-\alpha t / MTTF}$$

when

$$\alpha = \int_0^{\infty} [1 - G(t)] dF(t)$$

Is small.

Just by increasing the complexity of a network topology from one link to a double link the analytical reliability results have increased tremendously so that the formula for the double link is difficult to exploit.

5.2 SIMULATION

An investigation of the resiliency of replication control algorithms to topology and copy placement involves the examination of a large number of alternative systems. An environment in which to compare the algorithms with each other objectively is essential. Configurations need to be repeated with similar conditions in order to be certain that the comparisons are acceptable. Simulation offers an advantage over other approaches since it is an experimental technique,

which offers greater control over measurements and includes the following characteristics.

5.2.1 ADVANTAGES OF USING SIMULATION

Realism.

Simulation models can be realistic, in the sense that they capture the actual characteristics of the system being modelled.

Experimental control.

Every variable can be held constant except the ones whose influence is being studied. As a result the possible effect of uncontrolled variables on system behaviour need not be taken into account, as is often done when experiments are performed on real systems.

Reproducibility of events.

Although the elements of a system using simulation produce random behaviour with the use of random numbers, it is statistically predictable in that a particular set of events can be reproduced exactly if necessary.

Imaginary systems.

The systems whose behaviour is being investigated need not actually exist to be subject to simulation based experimentation.

Time reduction.

The equivalent of days, weeks, or months of real-system operation can often be simulated in seconds, minutes or hours on a computer. This means that relative to real-system experimentation, a large number of simulated alternatives can be investigated.

Deferred specification of objectives.

If the objectives of a system are not completely clear, i.e. the investigator is unsure which options are more important, using simulation gives room for open-minded experimentation into multi-criterion decision environments.

Chapter 6

The Simulation Environment And Experimentation

The choice of simulation environment for experimentation is vitally important. There is no one simulation environment that works equally well for all proposed simulation studies. Just as there are many diverse applications of simulation, so too are there a variety of alternative simulation modeling languages and software packages [4,87,100,107,108] and the opportunity to create a new simulation environment yourself that can be adopted to model specific requirements.

This chapter describes CLOWN the author's choice of simulation environment for this study. There is also an account of the simulation procedure and the process that has been used to accomplish the organization of the bulk of experiments.

6.1 CLOWN

The CLOWN (Concatenated LOcal-area and Wide-area Network) simulator [107,108] supports a modular architecture. It is a simulation based modeling tool developed at University College London that enables modeling of computer systems in a "realistic" and intuitive manner. The interface allows the user to construct systems using individual components and experiment with computer systems as though they were real. It is possible to simulate any network configuration that can be modeled by a collection of components, and observe the network topology visually. One is also able to modify the parameters or variables of the simulation experiment using a mouse.

6.1.1 MARKOV CHAINS AND STATE TABLES

The CLOWN model is based on an event-driven Markov chain with segmented state table. This technique can be used when it is possible to establish an exclusive and complete mapping between events and state variables.

The segmentation process [108] is illustrated in Fig 6.1 which shows a model represented by its state table and a list of possible events. Each event may potentially cause a change in the state table, but will normally only influence the value of a specific limited set of state variables. In Fig 6.1 consider the set S_1 and a corresponding set of events E1. Provided that the members of E_1 operate only on S_1 and that all events which operate on *E1* are members of S_1, E_1 forms a segment. A model may contain several copies of a segment. The event set E2 will operate on both t S_{2a} and t S_{2b}, both of which form independent segments. In this example the model consists of 4 segments of which 2 are identical.

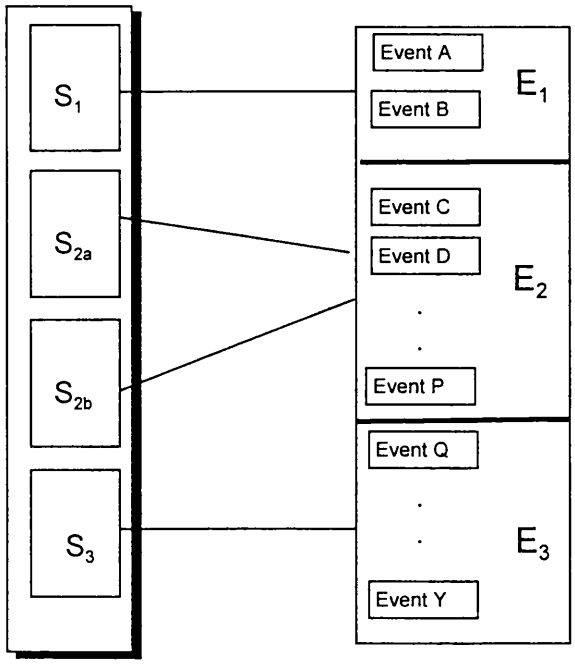


Figure 6. 1 Segmentation of a state table

On the basis of this segmentation exercise, it is possible to partition the model into a collection of self-contained objects each with their own state table and a set of methods to operate on them. The example above contains four segments but only three objects. The segments S_{2a} and S_{2b} are not separate objects. Instead they represent separate instants of the same objects. Each instance of an object has full control over its own state table, which can not be amended by an outside entity. The only way to change the state of an object is by issuing a request in the form of a message to the target object. As the name indicates, these messages do not themselves trigger a change of state. They merely request the object to evaluate whether a change in the state is appropriate and perhaps, in turn cause a request to another object to be issued. In the case of CLOWN, the requests map directly onto the model events.

For example, one can define a module as having two states: "available" and "unavailable". At any given point in time thestate table will represent the state of the module in relation to the system. The set of possible events will be "failure" and "recovery" events. When a "failure" event is generated, a request that consequently changes the state to "unavailable" is initiated, and when a "recovery" event is generated the resulting state changes to "available".

6.1.2 EVENTS

An event does not signify a change, it acts as a marker indicating that it is time to re evaluate the state of a particular state table segment. This is associated with the fact that the Markov chain has no memory and will therefore reevaluate the situation every event time, before taking the relevant action. A particular object in a CLOWN model can exhibit stochastic behaviour. For example, a module's "running time until its next failure" may be statistically predictable by assuming that it is exponentially distributed with a mean of 20 days, events can be generated to represent its "failure".

6.1.3 LANGUAGE

CLOWN does not implement a special simulation language, but uses ordinary C code.

6.1.4 FAILURE AND RECOVERY OF COMPONENTS

The CLOWN simulation programme allows the components to parameterize their failure and recovery distributions. The programme supports a large variety of distributions including the deterministic, i.e. one is able to decide the exact time that a component fails or recovers.

6.2 SIMULATION ENVIRONMENT AND COMPONENTS

6.2.1 MODELING

The experimental model is a simplification of the real environment. It presents us with an opportunity to investigate actual systems as simplified and controlled as we wish in order to achieve a better understanding of particular or specific concepts. We may wish to disable and enable some of its properties, use the system in fundamental forms or simply run it as it is. The flexibility offered by modeling may never be achieved by experimentation on actual systems simply because it is not practical or cost effective to alter the configuration of the system at will, simply in order to understand it better.

Two important phases of good modeling are validation and verification.

Validation

A good "model" of an actual system is an abstract representation of it. The concept of the model must be validated to ensure that it behaves or performs like the actual system or as close to it as possible. If one does not have a good understanding of the concepts of the actual system then one cannot "model" it in a manner that will produce results that can be used to make inferences or conclusions about it.

Once the model is validated one may assume it is a good model and is representative of the actual system.

Verification

The "program" or apparatus that is being used to evaluate the model of the actual system must be verified to ensure that it exemplifies the model concept as has been

validated. One must ensure that the model concepts are represented by the apparatus used, since it is not enough to understand the model, but it must be built to behave as expected.

6.2.2 THE SIMULATION EXPERIMENTS

The simulation experiments evolved in three stages The first stage was developmental and involved the design and creation of the environment for this study. This consisted of the design of the components of the systems and the organization and implementation of the replication control algorithms. Themodel concept was determined.

The second stage was explorational, it consisted of experimenting with this environment and the components in order to establish an understanding of how they work. This stage can be divided into a comprehension period, followed by validation and verification of the model.

It was immediately apparent that a satisfactory investigation of a set of simulations to study the resiliency of replication algorithms would require a large number of experiments. In this phase the experiments were designed, methods were devised of how best to run simulations and organise the vast amounts of results that would be generated.

The third stage was practical and involved setting up simulation runs, executing them and organising their results.

6.3 *STAGE1*: DEVELOPMENT OF THE CLOWN ENVIRONMENT COMPONENTS

The aim in designing a simulated model of the distributed system (as defined in chapter 2) was to provide a representation that is equivalent to the actual system in all important aspects. The simulation environment and components were therefore designed with this aim in mind.

Three separate version of the simulation programme had to be implemented corresponding to the three replication control algorithms chosen for this study, namely Weighted Voting (WV), Voting With Witnesses (VWW), and Voting With Regeneration (VWR). Each version contained the same components with which to construct a network configuration.

The replication algorithms protocol was organised by a controller module that was not manipulated visually but programmed to run as part of the simulation. This module was also responsible for network routing, and general checking and validation of input data.

Network systems are assembled from four basic components that correspond to the definition of a distributed system given in Chapter 2: subnet, node, link, and an observer module that collects reliability data. The node is the only component that varies across the different implementations of the simulation.

6.3.1 WEIGHTED VOTING FOR SIMULATION

Once the read and write quorums are chosen, the program validates that the read quorum plus the write quorum are more than the total number of weights allocated to the copies.

At each event that occurs when a node or link fails or recovers, the controller module goes through the following procedure:

1. A routing table is created which records whether there is a route from any component to any other.

2. For each observer the routing table is searched to see whether there is a route from the subnet to which it belongs and all the other subnets in the network. If there is a route then each copy on these subnets is checked to see if it is available and the weight for this copy is recorded.

3. The controller module then determines whether the system is read-available and write-available at this moment in time (according to the different read and write quorums). This information is collected in read-reliability, and write-reliability tables.

A copy that has failed is brought up-to-date upon recovery. This basically means that an infinite update rate is presumed.

6.3.2 VOTING WITH WITNESSES FOR SIMULATION

In addition to the processing performed for the WV (above), each observer module ensures that a quorum has at least one real copy and that its version number is equal to or greater than the highest version number of a witness included in the same quorum. This validation must be performed since witnesses have no data, therefore a quorum that has no "real" copies, or copies that are not up-to-date is not acceptable.

6.3.3 VOTING WITH REGENERATION FOR SIMULATION

Each node is given an order number to represent the order in which the system should find a new node if a node becomes unavailable. In addition to the validation performed for the WV, if a quorum is not met then the system attempts to regenerate a new copy on an available node with the highest order number as follows:

1. For each observer view, the controller module examines all the nodes with copies to find out whether they are available or not.

2. If the node being examined is available then the observer records its weight and processing continues.

3. If the node is unavailable then the spare nodes (without copies) are inspected. If there are available spare nodes then the node with the highest order number is chosen. The flag indicating whether this node has a copy is set to on,

conversely, the flag on the unavailable node is set to be a spare (an unavailable spare).

6.3.4 NODE

This component represents the node in a distributed system containing the "copy". It fails and recovers according to a distribution function and has only two states; up or down. When required this module may be given the "perfect" status i.e. always up, by selecting its failure distribution as deterministic with the failure time higher than the length of the simulation run.

All versions of this component have a weight associated with them. The differences according to which replication algorithm is implemented are:

6.3.5 WV

A node does not contain a version number since the system assumes that as soon as a node has recovered it has the correct version of the copy.

6.3.6 VWW

A node for VWW has a tag classifying whether a node contains a "real copy" or a "witness". There is also a version number associated with it because finding a quorum does not necessarily mean that the update can be accepted. There must be an actual copy with the highest version included in the quorum, since a witness does not contain any real data.

6.3.7 VWR

A node for VWR also has a tag specifying whether the node has a "copy" or whether it is a "spare" node. It also holds the order number that is associated with the regeneration of copies. This module does not need a version number since as soon as a node recovers it is assumed that it is bought up-to-date.

6.3.8 SUBNET

This component does not fail. Nodes that belong to a subnet are fully connected, although each node may fail individually. The advantage of having this module is to represent a connection point between nodes and links that highlights the different types of failure. It is then easy to differentiate between node failure and partitioning, since a system can only partition when links are down. Considering a system this way does not make any assumptions about how often partitioning occurs but relies on the fact that links may fail and recover.

6.3.9 LINK

The connection between subnets is provided by this component. Each link is bi directional, and there are no failures that cause the link to function unidirectionaly. Like the node, it has two states: up or down. It fails and recovers according to a distribution function.

When required the link may be made "perfect" by giving it a fixed time to failure greater than the length of the simulation run.

6.3.10 OBSERVER

An observer is placed in every position in the topology that may produce a different reliability function. Whenever there is a failure (node or link) or recovery event each observer collects reliability data that is passed to it by the controller module. Each observer therefore has its own view of the reliability function.

6.3.11 THE RELIABILITY DATA

We understand from section 2.6.2 that the reliability function resembles a negative exponential. For this function, any uncertainty in determining data points near the beginning and in its tail will have significant impact on its shape. There is consequently a case for collecting more data points in these areas. The reliability data was collected at multiples of a fixed time interval which is referred to as the *reliability unit, t*. The first being 0*t* (the availability), *t, 2t, 3t, etc.* The reliability unit was chosen to be small enough to get an accurate integral, without being so small that you had to collect enormous amounts of data.

Using values *R(0)*, *R(t)*, *R(2t)*,.... the reliability function is approximated (explained bellow).

During a simulation run, if a read or write quorum cannot be collected, the view of the system from a particular observer is unavailable (otherwise it is available). Each observer then has a collection of time periods in which the object was available during the total simulation run.

The availability for this view is the total time of available periods divided by the time of the simulation run.

Using time unit, *t*, as the reliability unit, each period of availability is examined for continuous availability for a period of δ time units. These continuous time periods are then collected. Their total over the total simulation run time is the next data result $R(\delta)$. Every time an object becomes either available or unavailable (to a particular observer) the time is noted. When it becomes unavailable to that observer, the time since it last became available, *d* say, is compared to the multiples of the reliability unit, 0, *t*, 2*t*, etc. for every n with $nt < d$, a count for that multiple is updated:

$$R_c[n] = R_c[n] + d - nt$$

$R(nt)$ can be found at the end by dividing by the simulation run time.

6.4 STAGE 2: EXPLORATION OF THE SIMULATION ENVIRONMENT

After programming the simulation environment and creating the components with which to assemble a network topology, some investigation time was apportioned to experimenting with the simulation. This time was used to establish an intuitive feel for its processing as well as to determine if there were any variables that did not make a difference to the reliability results. This stage of the investigation was also used for the validation of the simulation experiments.

Initially, it was exciting to experiment visually with different network topologies and alter the parameters spontaneously. It was easy to experiment with different configurations and various lengths of run time. However, once the bulk of the experiments were planned, it was tedious to have to set them up (hundreds of times) and run them individually. This experimentation stage also revealed that the investigation would have to be concentrated in order to achieve the objectives.

The primary difficulties in conducting a comprehensive set of experiments are the number of parameters involved even for small models and the organisation and analysis of the vast quantity of results. The experiments needed to be rationalized and normalized.

A simulation run involves two types of parameters; the simulation environment parameters and the component parameters. The simulation environment parameters apply to all the experiments performed on a particular network topology whereas the component parameters modify the characteristics of a particular component (e.g. a node or a link). A central feature in the organization of the experiments is the choice of parameter values. For instance the length of simulation run, the weight given to a copy or the failure distribution assigned to a component.

6.5 THE SIMULATION ENVIRONMENT PARAMETERS

This section describes the variables involved in setting up one simulation run. These variables do not involve the assembly of the network topology to be studied, but only the simulation experiment itself. Explanations for the author's choices for some of these variables are also outlined.

6.5.1 READ AND WRITE QUORUMS

The underlying algorithm that is implemented for a particular set of experiments is programmed into the simulation program. All the algorithms selected for this

investigation are classed as voting algorithms, therefore they all required read and write quorums.

6.5.2 THE SEED

The component failure and repair times are based on random numbers, therefore each experiment must be run more than once with a different seed each time in order to obtain a statistically significant average for each reliability value. A set of experiments for different topologies and copy placements were chosen to investigate how many seeds would required to obtain significant results. It was discovered that the variance when using three seeds or more was significantly small that it a decision was made to run each experiment with three different seeds.

6.5.3 LENGTH OF SIMULATION RUN

In order to determine the length of time that each algorithm-dependent experiment needed to run, a set of topologies that represented a crosssection of the topologies for experimentation were selected. Using this set of topologies the simulation was performed for increasing lengths of time. The reliability values were recorded for each of these times.

When there is no change in the reliability values between one particular run time and the following increase in time, the network system is has reached a steady state. It was immediately noticeable that the larger and more complicated a topology was, the longer it took to reach the steady state. The adopted run time was therefore the length of time that it took for the most complicated network topology to stabilize.

6.5.4 RELIABILITY UNIT

The collections of data for the reliability function at equal intervals of time are termed the *reliability units* for the purposes of this thesis.

A smaller reliability unit should generate a closer and smoother approximation to the reliability function. However, the value of the unit had to be balanced against the

amount of data that the program produced, since for each topology this reliability data is generated for each observation or subnet view.

Based on some experimentation with different reliability units, it was deided that a value for the reliability unit that was 5% of the distribution mean was sufficiently small to represent the reliability function well.

6.5.5 NUMBER OF RELIABILITY UNITS

The number of reliability units should be high enough to allow the last reliability value in the tail of the reliability function to approach zero. Before each set of experiments on a particular topology, a set of simulation experiments with different copy placements was performed to determine the number of reliability units needed in order to reach this state. The time needed to reach this state differed for various network configurations, but in order to ensure the reliability function approached zero, each experiment was run for three times the amount indicated by the initial experimental set.

6.6 COMPONENT PARAMETERS

This section describes the variables that apply to nodes and links and explains some of the decisions that were made regarding their values.

The subnet module remained constant. The only difference in the observermodules was a unique identification number that indicated from which observation point the results from particular view were obtained. This differentiation was necessary because the observer modules are identical.

6.6.1 COMPONENT FAILURE

It is often assumed that the failure and repair rates of components are exponentially distributed. There are also some investigations that assume that the repair rate is normally distributed. Long and Carroll [76] conducted an investigation to test the hypothesis that component failure and repair rates are exponentially distributed. Some of the samples were found to have a realistic chance of being drawn from an

exponential distribution, while others can be classed as nonexponential. The assumption that the failure and repair rates of components are exponentially distributed could not be contradicted. Therefore, for the sake of continuity with the literature, it is assume that the failure rates of components are exponentially distributed.

The only components that fail and recover in the simulation are the nodes and links. Due to time constraints it is assumed that all nodes are identical and all links are also identical. Each component may only be in a functioning or nonfunctioning condition. Failures occur independently of each other with exponential probability distribution $e^{-\lambda t}$ where λ is the failure rate of the component.

6.6.2 COMPONENT REPAIR

Investigations that have simulated failure and repair assume that failure is exponentially distributed. However, the repair of nodes is assumed to be exponentially distributed in some works, and normally distributed in others.

Two sets of experiments were performed to determine which of these assumptions would be preferable to this investigation. In both sets topologies were selected and run with an exponential failure distribution with mean values of (100,200,300). For one set the repair mean was assumed to be normally distributed with the mean values of (20,40,60) and corresponding standard deviations of (4,5,6). For the other set the repair times were assumed to be exponentially distributed.

The variance of the results of these experiments was not significant. It was therefore decided that the repair distribution is not significant and the mean time to repair was assumed to be exponentially distributed $e^{-\mu t}$ where μ is the repair rate. The exponential distribution was selected because it is easy to use.

6.6.3 FAILURE AND REPAIR RATE VARIATION

The availability of any component can be determined using the following formula:

$$A = \frac{\lambda}{\lambda+\mu} \qquad (6.1)$$

This equation can be used to vary the availability of a component by altering the values of the failure and/or repair rates. The simplest way to accomplish this is to alter only one of these i.e. either the failure or the repair rate. For example a node is assigned a failure rate of 0.05 and a repair rate of 0.2 for an availability of 0.8. In order to increase the availability to 0.9 one could either alter the repair rate to 0.45 leaving the failure rate constant or one could alter the failure rate to 0.022 leaving the repair rate constant.

A comparison of the reliability data for both these methods of variation, using the same topologies and availability values was analyzed using the area ratios (Chapter 7). As an example of this study consider the following topologies:

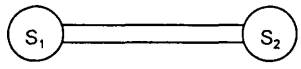


Figure 6. 2 Double link network

1. Double link network. Which consists of two nodes (s_1, s_2) connected together by two identical links. Where s_1 contains an object to be accessed.

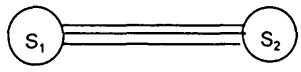


Figure 6. 3 Triple link network

2. Triple link network. Which consists of two nodes (s_1, s_2) connected together by three identical links. Where s_1 contains an object to be accessed.

The areas of the reliability functions, as viewed from s_2, for nodes of different availabilities were calculated for both topologies. The values of the areas for equivalent availabilities were different for the MTTF and MTTR. However, since we are considering the concept of relative reliability, these same areas were examined relative to the area of one link (i.e. the ratio of the area of one value over the area of one link) for equal availabilities. The results were exactly the same when varying the MTTF and

MTTR. Figure 6.4 illustrates this comparison for failure rate variation and Figure 6.5 illustrates this for repair rate variation.

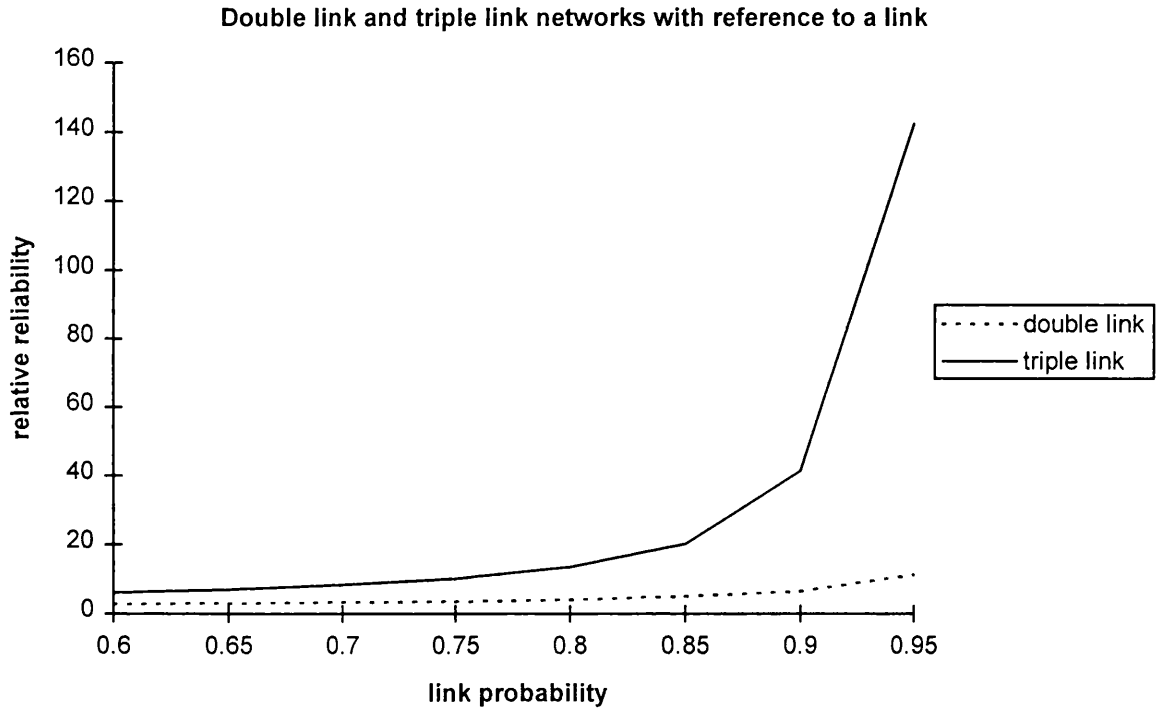


Figure 6. 4 Relative reliability with reference to a link varying the MTTF

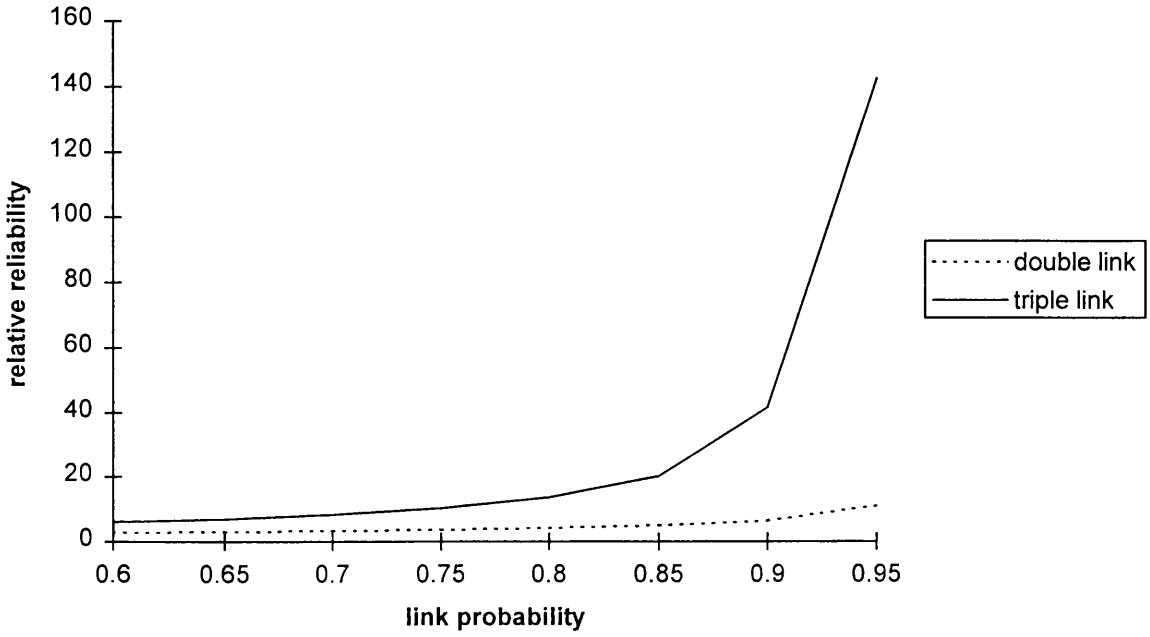


Figure 6. 5 Relative reliability w ith reference to a link varying the MTTR

A decision was made for experiments that required availability variation for their components; the repair rate was altered leaving the failure rate constant.

6.6.4 AVAILABILITY VARIATION

A series of experiments to understand the effect of increasing or decreasing the availability of components was performed.

The results of reliability data are illustrates in figure 6.6:

1. higher availabilities produced higher reliabilities

2. the reliability functions of varying availabilities follow exactly the same pattern

3. higher probabilities (e.g. 0.95) generated high reliabilities for the initial time units (e.g. *R(0) = 0.99984 and R(2)=0.99912*)

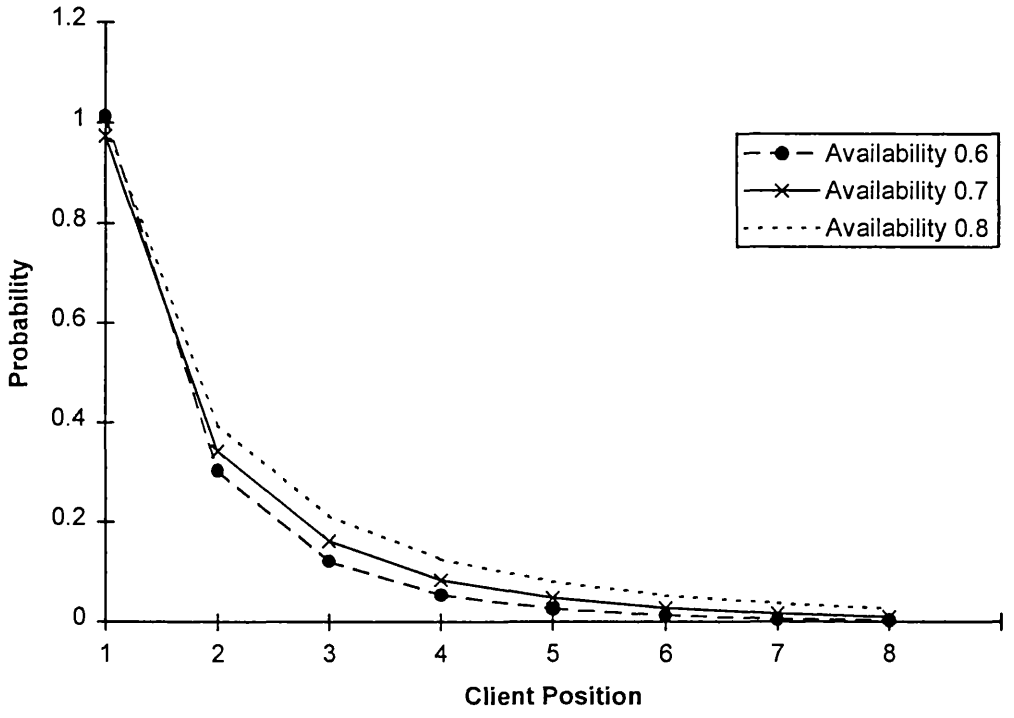


Figure 6. 6 Several Availabilities for network in a chain

The fact that varying the availability produced very similar reliability functions enabled the decision to perform the bulk of experiments using one value for the availability of components. In order to avoid extra reliability data, the availability of components was chosen to be 0.8. This was calculated using equation 6.1 with a failure rate of 0.05, a value based on the fact that many of the machines tested in a study into the mean time to failure of machines [76] seem to have a mean time to failure of approximately 20 days.

6.6.5 WEIGHT GIVEN TO NODES

Since all of the algorithms implemented for the experiments are voting techniques, each copy requires the allocation of a weight. This allocation may affect the performance of the algorithm, however, due to time constraints all the copies were given equal weights of 1.

6.7 VALIDATION

The calculation of reliability analytically is extremely complicated if not impossible in most cases. This means that the validation of the simulation has to rely mostly on analytical availability results.

6.7.1 AVAILABILITY OF A SINGLE LINK

The availability of a single link can be validated by simulating a topology that consists of two subnets connected together by a single link. One subnet contains a "perfect" node (a node that does not fail) and the other contains the observer. Assuming that the link fails and recovers with an exponential probability distribution with λ failure rate and μ repair rate. Using the equation 6.1 above availability can be determined analytically and compared with the simulation results for equivalent failure and repair rates.

Simulated Availability	Analytical Availability
0.3964	0.4
0.4963	0.5
0.5975	0.6
0.6971	0.7
0.7978	0.8
0.8987	0.9

Table 6. 1 Availability of single link

6.7.2 AVAILABILITY OF A DOUBLE LINK

A double link can be represented by two subnets, one containing a perfect node, and the other containing an observer, connected together by two links. If the availability of a link is *A*, then the analytical availability of both links is $12(1-A)^2$.

Analytical Availability	Simulated Availability
0.64	0.6362
0.75	0.7510
0.84	0.8390
0.91	0.9105
0.96	0.9611
0.99	0.9896

Table 6. 2 Availability of double link

6.7.3 RELIABILITY OF NETWORK CONNECTED IN SERIES

As shown in section 5.1.4 the analytical results for the reliability of a network connected in series can be calculated using equation:

$$R(t) = A^n e^{-n\lambda t}$$

It is possible to fit an exponential distribution to the simulation results of a network connected in series. This is done using weighted regression. The largest weight is assigned to R(0) and is decreased by a factor of two for each of the following reliability units. Figure 6.7 illustrates an analytical reliability and its approximation using the simulated results.

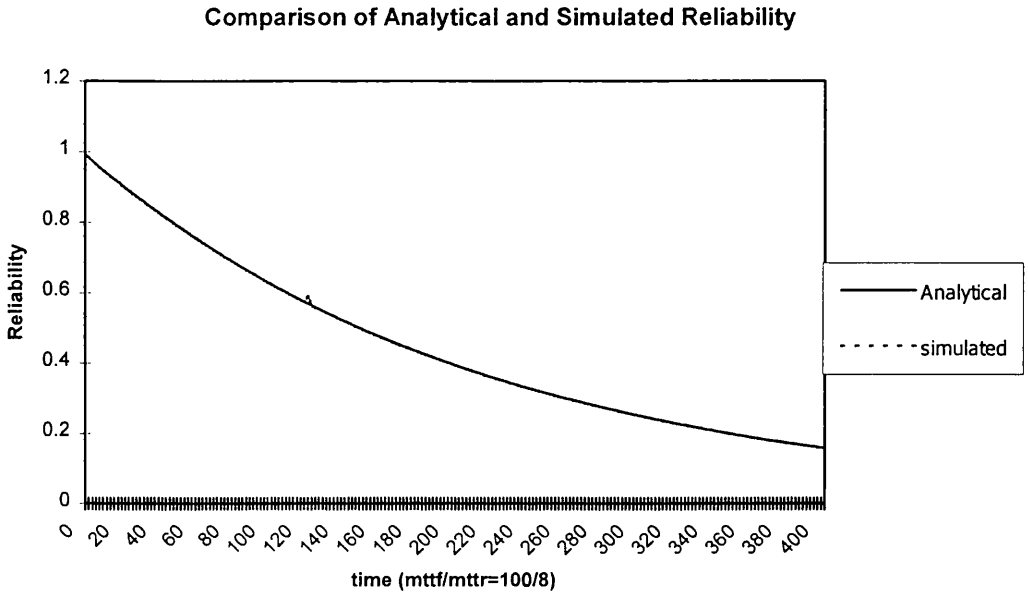


Figure 6. 7 Comparison of analytical reliability and simulated reliability functions

6.7.4 PARIS'S ANALYSIS OF WEIGHTED VOTING AND WITNESSES

Pâris's [87] presented comparison of the availabilities of WV with three copies and VWW with two copies and one witness. He assumes that the probability that a given site will experience no failure during a time interval of duration t will be given by $e^{\lambda t}$, where λ is the failure rate. His results are ideal for validation purposes, since the simulation experiments in this thesis include an implementation of both WV and VWW.

The assumption in this paper were therefore used in a setof simulation experiments and compared to the analytical reliability results.

Figure 6.8 illustrates the simulated results of the analytical results in this paper.

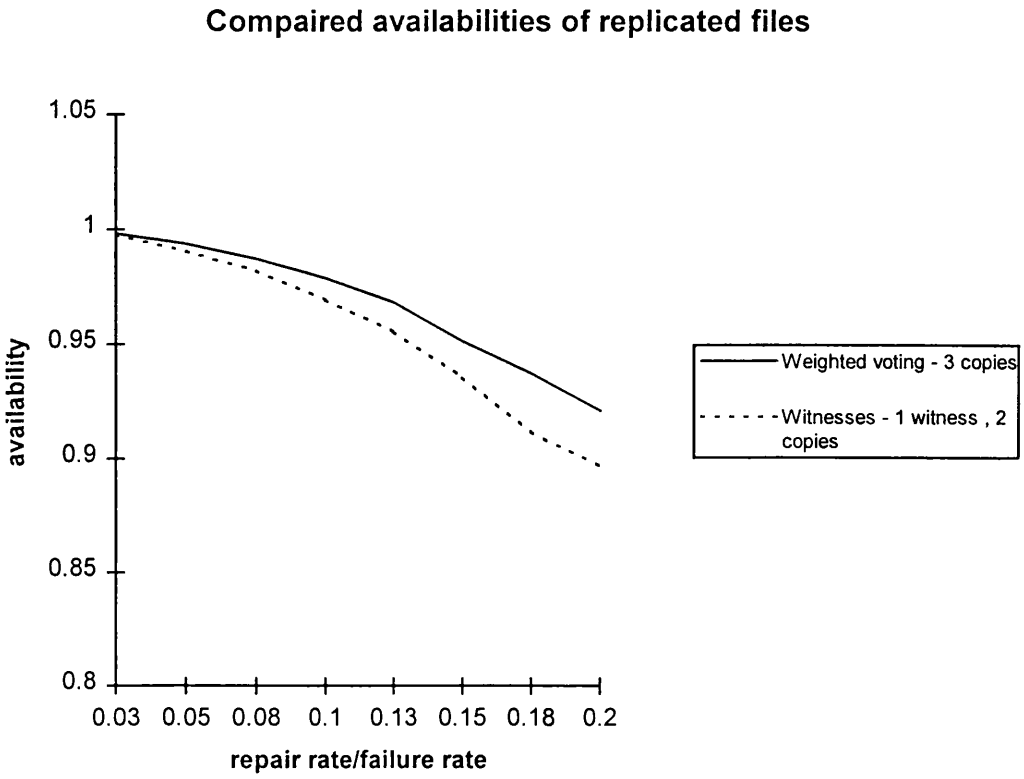


Figure 6. 8 Simulated results for the Compared Availabilities of Replicated files

6.7.5 DISCUSSION

The validation section demonstrates that the simulation experiments are performing appropriately. We have compared the simulation results to the limited but accepted analytical reliability results and they have proved to be consistent with the literature. The analytical results are calculated for basic network configuration. They do not include copy placements and the network topology since it is very difficult (and in some cases impossible) to calculate analytical reliability results.

6.8 STAGE 3: THE EXPERIMENTS

The aim of the experiments is to show how a comparative reliability measure can be used to study the resiliency of replication algorithms. This investigation examines the resiliency of three replication algorithms to changes in network configuration and copy placement.

6.8.1 EXPERIMENTAL PLAN

The explorational stage of this study revealed that a competent investigation of the resiliency of replication algorithms necessitated a large number of experiments. Mainly because a thorough study of copy placements means the examination of all the possible permutations of copy placements within any topology. Furthermore each set of experiments into a particular topology would have to be executed three times so that any particular copy placement could be compared across the three simulated replication algorithms.

The first phase of the experimental plan is therefore the decision of which topologies to investigate. How many copies to consider for the replicated object comprises the second phase. The third and final phase involves the strategy employed to implement the decisions made in the earlier two phases.

6.8.2 NETWORK TOPOLOGIES

A study that involves changes to the network configuration must entail different network topologies. This is a difficult issue to address since there are so many topologies one can examine and ways in which they can be altered. It is essential, in this study, to examine whether there are specific topologies that enhance or degrade the performance of the concurrency control technique being used.

An inspection of network configurations led to the conviction that using a distributed system (as defined in chapter 2) any arbitrary network configuration can be constructed from any of the following basic network topologies:

6.8.2.1 CHAIN

S_1 is connected by a link l_1 to S_2, which is connected by another link l_2 to S_3, which is connected by another link l_3 to S_4 etc. Figure 6.9 illustrates a five-subnet chain topology.

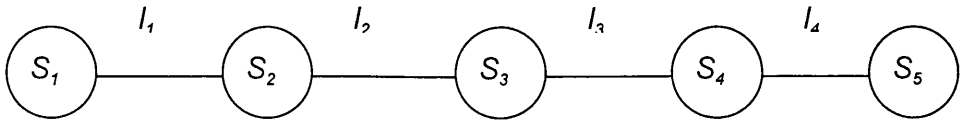


Figure 6. 9 Example of a chain topology

Consider an object that is placed on a node in subnet S_1 then a request to access this object from S_4 must find l_3 , l_2 and l_1 and available as well as the node containing the object, before the request is accepted.

6.8.2.2 RING

S_1 is connected by a link l_1 to S_2 which is connected by another link l_2 to S_3 which is connected by another link l_3 to S_4 etc. If there are n subnets in the topology, then S_{n-1} is connected by link l_{n-1} to S_n which is in turn connected to S_1 by link l_n. As its name implies this topology forms a full circle, so that there are always two ways of approaching every subnet. Fig 6.10 illustrates this topology for five subnets.

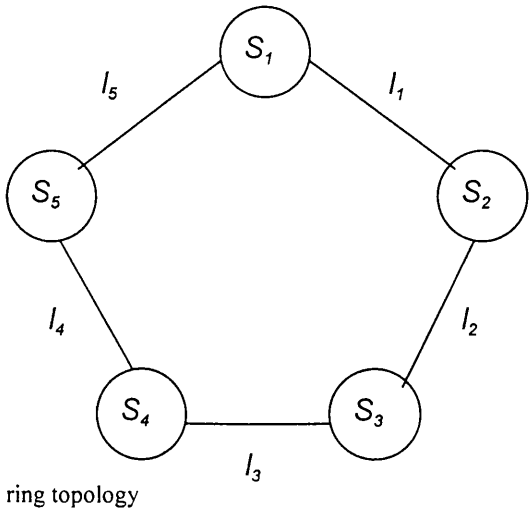


Figure 6. 10 Example of a ring topology

It is easy to observe that this topology can also be formed from a connected chain.

6.8.2.3 STAR

S_2 , S_3 , S_4 , ... are all connected by a single link to S_1 . Therefore S_1 is at the center and has several links connecting it to other subnets. Fig 6.11 presents a fivesubnet star.

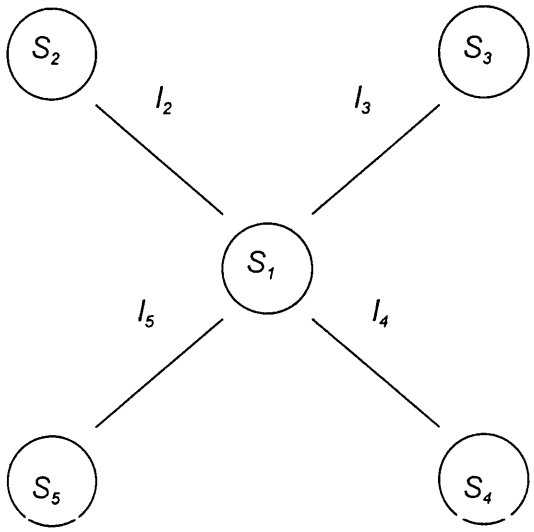


Figure 6. 11 Example of a star topology

Consider an object placed on S_1, a request to access this object from S_4 requires l_4 to be available as well as the node containing the object (S_1). However, if the node containing the object is unavailable then the object cannot be accessed from any position in the network!

These topologies must be studied in order to understand the impact of network configuration on the reliability.

A survey was conducted of the network topologies of ten local establishments that maintain distributed systems. Most of these establishments developed topologies

according to their building and departmental organization that could be constructed with one or more of the three fundamental topologies outlined above.

A study of network configurations based on these three has provided an insight to the performance of concurrency controltechniques under any topological situation.

6.8.3 EXPERIMENTAL MODEL

The experiments were based on all the possible combinations of five subnets connected together. All the possible placement of three copies within these subnets were examined for each replication algorithm. More specifically for WV there are three copies, for VWW there are two copies and one witness, and for regeneration there are three copies and two spares.

All the nodes and links have an availability of 0.8 unless otherwise stated. Thereis an observer module on every subnet.

6.8.4 CALCULATING THE AREA

The analysis of the simulation results is performed using the area ratios. Simpson's rule was used to calculate the area under the reliability function. The tail end of the curve was calculated using interpolation with the assumption that the tail resembles that of a negative exponential.

The value of the area is not used directly but compared against the area of the reliability function of a non-replicated object with an availability of 0.8 In cases where a different reference is used it will be stated.

6.9 SUMMARY

The experimentation process was conducted in three stages: developmental, explorational, and experimental. The first stage involves the development of the environment and the components with which to construct network topologies. The second stage consists of a period of experimentation followed by the validation of the

simulation. The third stage involves the actual execution of a set of experiments to investigate the resiliency of replication algorithms.

Chapter 7
Analysis of Results

The analysis of the reliability results were consolidated into three sections pertaining to the categorisation of the basic topologies in Chapter 6, namely chain, ring and star. The impact of the network configuration on the reliability is examined in stages. First a reference value is established that can be used to relate the results to. Then network topologies for each of the chosen consistency control technique are examined for similar copy placements and scenarios.

7.1 ANALYSIS TECHNIQUE

This section describes the techniques and concepts used when analysing the results of the experiments. It is important to be aware of the parameters that are involved. This includes the type of node and link failures assessed, the copy placements and topology arrangements.

Each topology was examined with two scenarios that should present quite different results:

1. The nodes can fail and recover but the links are perfect.

2. Both the nodes and links can fail and recover.

The first of these situations is actually a study of the topology for the case where there are no partitioning failures. The node is either working and can be reached or not working and cannot be reached. This means that the reliability as viewed from different subnets becomes irrelevant since it is equal from each view in the network topology. The improvement in using replication for a particular topology can then be evaluated.

The analysis of network configurations that do not partition illustrates the reduction in performance due to partitioning and allows the evaluation of the reduction in reliability in systems that do partition.

The second scenario includes situations where partitioning can occur, since a subnet may be working, but a link to it is not.

There are numerous parameters involved in describing the results of these experiments:

- the topology; i.e. the arrangement of subnets and links;
- whether nodes and links fail and recover or are perfect;
- the placement of copies;
- the comparative value of each subnet view both in size (the size of circle) and in value;
- the replication algorithm used.

Examples from the results are illustrated using the presentation method described in Chapter 4 (section 4.4). In summary, hollow circles represent perfectly connected collections of nodes. Small, black circles represent the replicated object.

7.2 SUBNET VIEWS

The view of the reliability from a particular subnet is referred to as the *subnet view.* Hence, the subnet view from one subnet S_p can be compared to a subnet view from another subnet S_q . This means that reliability as viewed from different positions in any network configuration can be compare using the subnet views.

Under the scenario where links do not fail and recover and are "perfect", different subnet views become irrelevant, since they are all equal. This scenario is a good opportunity to

study how much more reliable one replication algorithm is compared to another without partitioning being present.

In Chapter 2 a subnet was defined as a fully connected set of nodes where the communication links between them never fail. In the case of a non-replicated object the view from any node in such a subnet is equivalent to the view of any other node as shown in Fig 7.1. For this example there is a deviation from the notation technique, in order to stress the point that this is an example of a fully connected set of nodes in one subnet, with a copy of an object on one of the nodes. The links are represented by lines, and the nodes are represented by a diamond with a n inside (representing a node). The subnet is the hollow circle. The number of nodes shown in this example (4) is irrelevant, since a subnet could have any number of nodes.

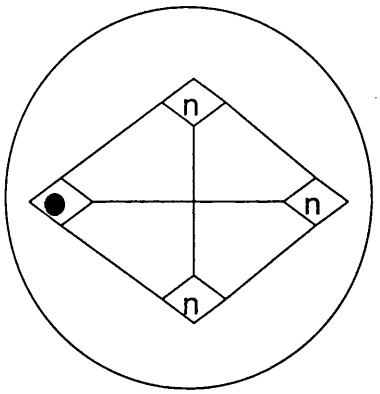


Figure 7. 1 A subnet with a non-replicated object

If the node with the copy fails then none of the nodes in the subnet are able to access the object including the node with the copy. Conversely, if the node with the copy is available then there is always a route from any node that is available to the copy since the links never fail.

7.2.1 AREA RESULTS OF A NON-REPLICATED OBJECT

In Chapter 4 (section 4.3.3) we calculated the analytical value of the area of reliability for each subnet view of a chain topology, assuming that the reliability curve was negative exponential. The area of the first subnet from the copy (namely, the value of the area for S2) gives us the area for a non-replicated object if the nodes are "perfect" but the links may fail (and recover). We assumed in section 4.3.3, that the nodes do not fail and the probability of the link being up is 0.8 with Mean Time To Failure of 20. In that case the analytical value of the area calculated in section 4.3.3 was 16.0.

The evaluation of the equivalent network configuration by simulation yields an area for a non-replicated object of 16.3. The reason for this discrepancy is attributed to the assumption inherited from the analytical evaluation that the reliability function was a negative exponential. Consequently, the actual simulation results have been replaced by the best-fit negative exponential distribution and as a result the analytical and experimental values differ slightly.

The area determined from this model is important is because it will be used in the following as a reference value with which to compare the area ratios of other configurations. The whole concept of comparative reliability is to find a suitable reference value with which to compare the area ratios. This value is considered an excellent reference point since it represents the starting point of all networks; one copy of the object not encumbered with replication or partitioning.

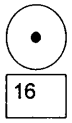


Figure 7.2 Reference Subnet

Fig 7.2 illustrates the reference point subnet and its corresponding value. All nodes in this system will be deemed to be inside the circle.

7.2.2 INCREASE IN RELIABILITY DUE TO REPLICATION

Consider the subnet illustrated in figure 7.1 with a replicated object, as shown in figure 7.3. This is the graphical interpretation of the definition of a subnet (definition 2.1) where a subnet represents a fully connected set of nodes such that the communication links connecting these nodes never fail.

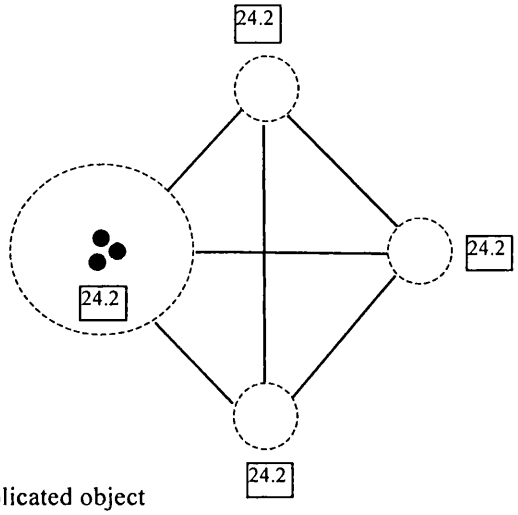


Figure 7. 3 Subnet with replicated object

The larger hollow circle represents the reliability of a replicated object as seen from anywhere in a network with perfect nodes (i.e. forming a single subnet).

The area value for this configuration is 24.2. This value was derived from the simulation experiments. We can use the reference value of the non-replicated case (as explained above) to quantify the increase in reliability due to replication. By using the area values 24.2/16.3= 1.5, we deduce that replication increases the reliability one and a half times. Hence the comparative reliability is 1.5.

7.3 CHAINS

A study of chains provides an opportunity to investigate the behaviour of the reliability as the distance from the majority of copies increases. We are expecting a reduction in the reliability as the distance from the majority of copies increases since more links and/or

nodes must be available the further the distance from the majority of copies. The reduction in reliability should be especially apparent when partitioning is considered.

7.3.1 NON-REPLICATED OBJECT ON A CHAIN

Consider a non-replicated object that is placed in a subnet where the nodes fail and recover with the same availability as the links (Fig 7.4).

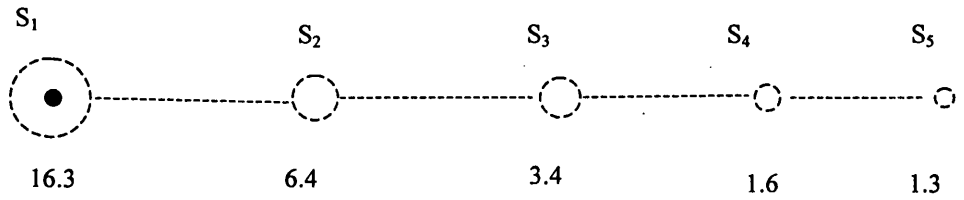


Figure 7. 4 Non-replicated object on a chain topology

The area value from S_1 is our reference value. The reduction in reliability in terms of link distance from an object follows a negative exponential function. Comparing the area values against our reference shows that comparative reliability as viewed from S_2 is 0.4 (6.4/16.3), from S_3 is 0.2 (3.4/16.3) etc.

7.3.2 PLACING COPIES ON A SINGLE SUBNET

Consider the same topology as the one in Figure 7.5, of a replicated object with three copies, where all copies are placed in S_1. The nodes fail and recover with the same availability as the links.

Since the consistency control technique must be included at this point, there are three cases to consider.

Weighted Voting (WV)

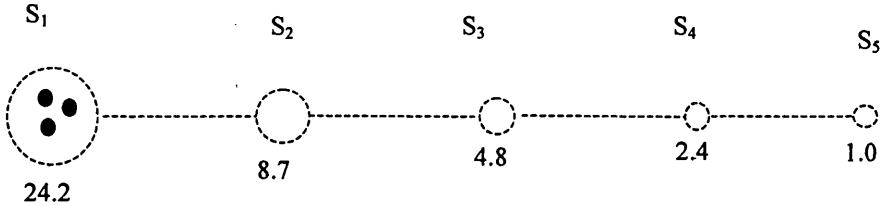


Figure 7. 5 Reliability with WV in a chain

In section 7.2.2 we showed that the increase in reliability compared to the non-replicated configuration without partitioning is 1.5. When one considers partitioning, we can compare the values of each of the subnet in both topologies as follows:

S_1	S_2	S_3	S_4	S_5
24.2/16.3=1.5	8.7/6.4=1.4	4.8/3.4=1.4	2.4/1.6=1.5	1.5/1.0=1.5

The increase in reliability is quite consistent for all the subnets.

Voting With Witnesses (VWW)

The comparative reliability of VWW is very similar to WV. The only variation is a small decrease in reliability for using a witness to replace a copy.

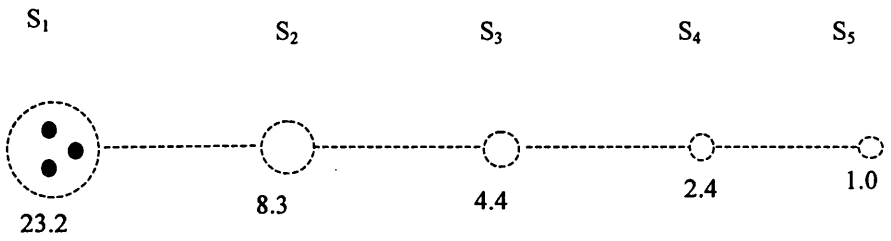


Figure 7. 6 Reliability with VWW in a chain

S_1	S_2	S_3	S_4	S_5
23.2/24.2=0.96	8.3/8.7=0.95	4.4/4.8=0.92	2.4/2.4=1	1.0/1.0=1

The decrease in the reliability in using a witness rather than a real copy is quite small (%0-%8). The difference is less visible as one gets further away from the majority of replicas.

Voting With Regeneration (VWR)

VWR is slightly more complicated since the position of spare nodes alters the reliability. For the case when there is one spare node and no partitioning, the area value is 76.5., when there are two spare nodes the area value is 221.4. This means that when there is no partitioning the increase in reliability with respect to the non-replicated object for one spare is (76.5/24.2) 3.2 and (221.4/24.2) 9.2 for two spares.

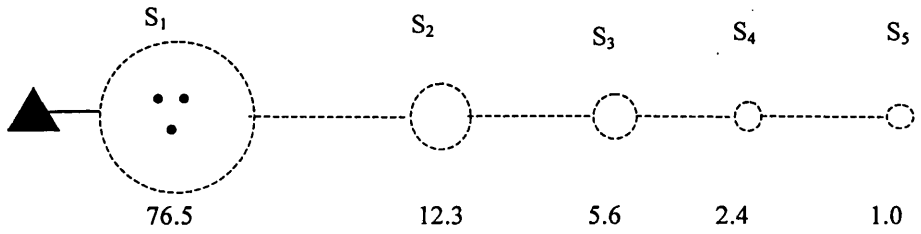


Figure 7. 7 Reliability with VWR with 1 spare, in a chain

Consider the topology in Figure 7.7, which illustrates the same topology as shown in Fig 7.5 and 7.6 but with a spare node on S_1. We may study the relative reliability as compared with WV. For one spare

	S_1	S_2	S_3	S_4	S_5
For one spare	76.5/24.2=3.2	12.3/8.7=1.4	5.6/4.8=1.2	2.4/2.4=1	1.0/1.0=1
For two spares	221.4/24.2=9.2	14.9/8.7= 1.7	6.6/4.8=1.4	3.2/2.4=1.3	1.3/1.0=1.3

It is interesting to note that regeneration with two spares introduces a significant increase. Reliability increases by a factor 9.2 when there is no partitioning, compared to an increase of 1.7-1.3 when partitioning is introduced.

If the position of the spares is altered the comparative reliability changes. For example consider three examples of this topology with three different spares positions

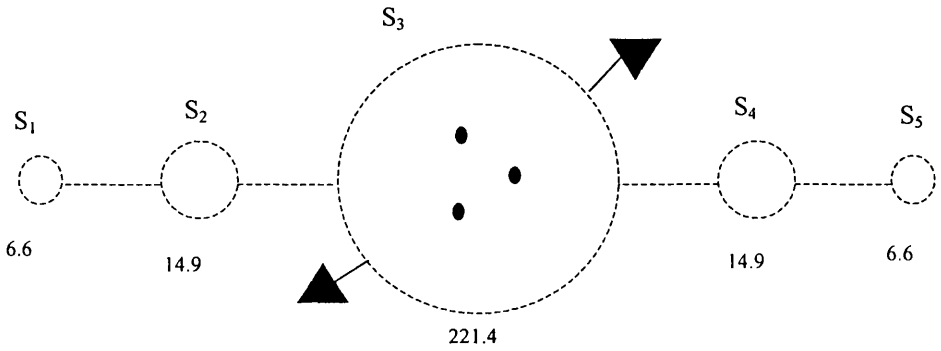


Figure 7. 1 Reliability with VWR and 2 spares, example 1 , in a chain
Figure 7. 8 Reliability with VWR and 2 spares, example 1, in a chain

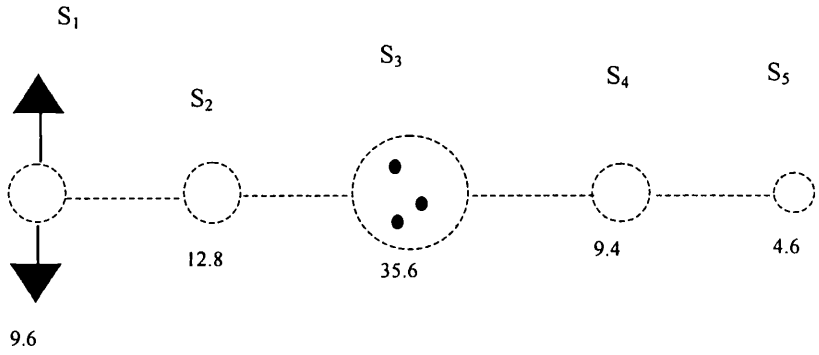


Figure 7. 9 Reliability with VWR and 2 spares, example 2, in a chain

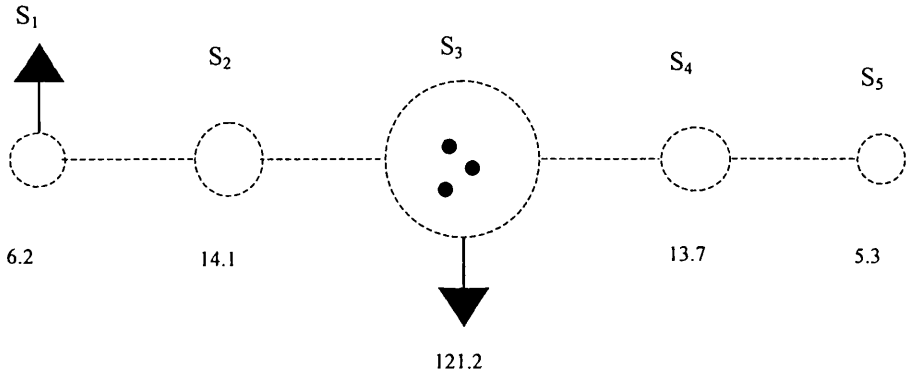


Figure 7. 10 Reliability with VWR and 2 spares, example 3, in a chain

It is easy to see that positioning the spares closer to the subnet containing the copies generates higher reliabilities. This is to be expected since the availability of the spares (when they are regenerated) is still dependent on the distance from the majority of the copies; the closer the regenerated copies are to the majority, the larger the increase in the reliability due to regeneration.

Discussion

The reliability in a chain decreases with the distance from the majority of the copies. The results of the experiments have corroborated that the decrease follows a negative exponential function.

We have also seen that there is a slight decrease in reliability when using VWW as compared to the WV. This decrease is due the replacement of one of the copies with a witness. As a consequence of the comparative reliability measure, we have been able to quantify that this decrease is not significant. This is exciting, since it illustrates that VWW is an excellent alternative to WV when cost considerations make it desirable to have two copies and a witness rather than three copies.

It has been visible in the literature that VWR is a very promising technique to apply to almost any algorithm. In this work we have been able to demonstrate not only that the increase in reliability when using VWR makes this consistency technique a very interesting one, but we have also quantified this increase. We have confirmed that the increase in reliability for the non-partitioning case is especially appealing. What is remarkable, is that the increase in reliability is very much diminished when partitioning is introduced.

7.3.3 DISTRIBUTING COPIES IN A CHAIN

The distribution of copies generates reduced reliabilities compared to a chain with copies in the same subnet. This is because collecting a majority in order to vote is dependent on the distance from the majority. The further the distance between the copies, the larger the decrease in reliability.

Consider the topologies in Fig 7.11. 7.12 and 7.13 for WV.

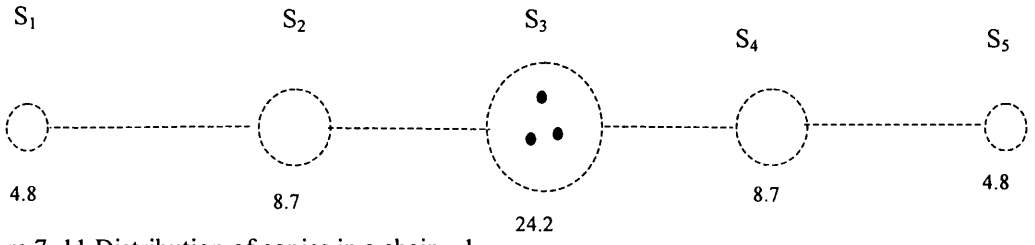


Figure 7. 11 Distribution of copies in a chain - 1

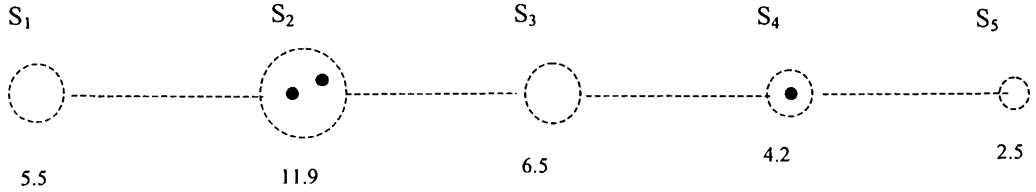


Figure 7. 12 Distribution of copies in a chain - 2

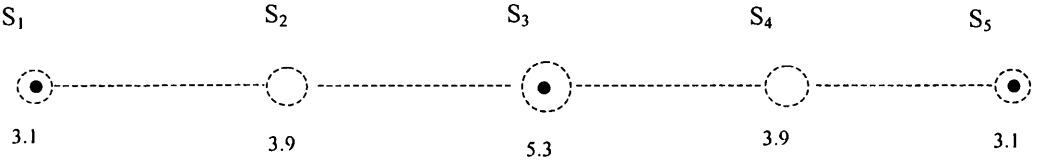


Figure 7. 13 Distribution of copies in a chain - 3

The topology in Figure 7.11 where the majority of the copies are kept together has higher area values than the topologies where the copies are distributed (Fig7.12 and 7.13). We can therefore make the following observations on the distribution of copies.

- The subnet containing the majority of copies has the highest comparative reliability.
- When copies are distributed, then for any two identical subnet views, higher reliabilities are obtained when additional copies are placed in adjacent subnets.

Voting With Witnesses

The comparative reliabilities for VWW are very similar to the reliabilities for WV. The only variation occurs in the positioning of the witness in relation to the copies. For example, consider figure 7.14 and 7.15

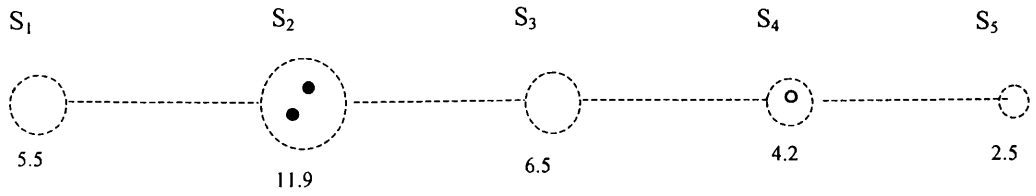


Figure 7. 14 VWW distribution of copies and witnesses - 1

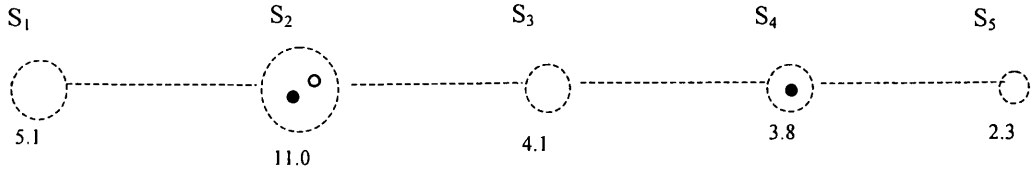


Figure 7. 15 VWW distribution of copies and witnesses - 2

There are two identical topologies with similar distribution of replicas. In Fig 7.14 subnet S_2 has two copies whereas subnet S_4 has a witness. On the other hand in Fig 7.15, subnet S_2 has one copy and one witness, whereas subnet S_4 has one real copy. The topology in Fig 7.14 has slightly higher values. For any two identical chain topologies with similar copy placements, higher comparative reliabilities are attained if the copies are kept together or are as close as possible to each other. This is because there is a higher probability of obtaining a majority that includes at least one real copy when both copies are together or closer together. We can compare the area values in Fig 7.14 and 7.15 as follows:

S_1	S_2	S_3	S_4	S_5
5.5/5.1= 1.1	11.9/11.0=1.1	6.5/4.1=1.6	4.2/3.8=1.1	2.5/2.3=1.1

There is an increase in the reliability for each subnet view, especially for S_3. The arrangement of copies and witness is meaningful. It is important therefore, to keep the copies together and distribute the witnesses, if one is distributing replicas. When deciding to implement VWW in order to reduce the cost of storing another replica, it is

important to consider the positioning of the witnesses in relation to the copies in order to reduce the degradation in the reliability compared to the WV technique.

It is Important to note at this point, that since replication control algorithms involve quite a lot of inter-copy communication, the reliability is significantly affected by having unreliable links between copies. Other factors that affect copy placement (keeping them physically apart to avoid them being affected by environmental problems e.g. flooding), faster user access by placing them close to user activity and so on, are not at odds with the need to keep copies close for improved reliability. It is also significant to remember that keeping copies close together does not mean on the same host since we are considering subnets.

Voting With Regeneration

In distributing the copies with regeneration, generally, higher reliabilities are generated when the spares are kept together, or as close as possible to the majority of the copies.

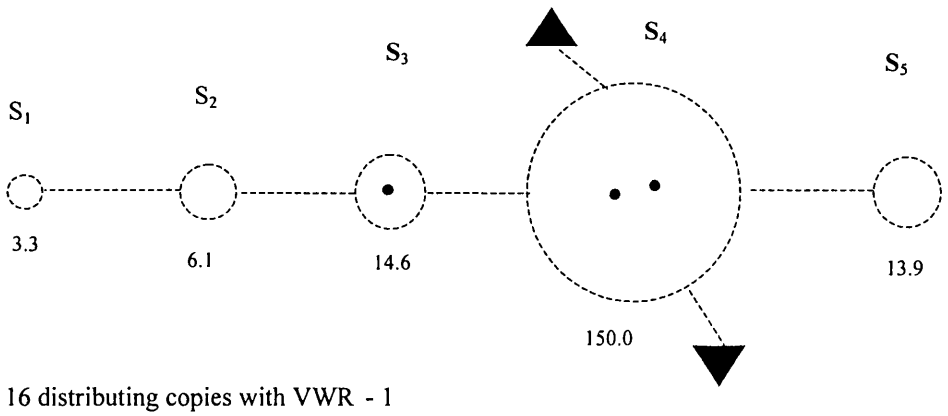


Figure 7. 16 distributing copies with VWR - 1

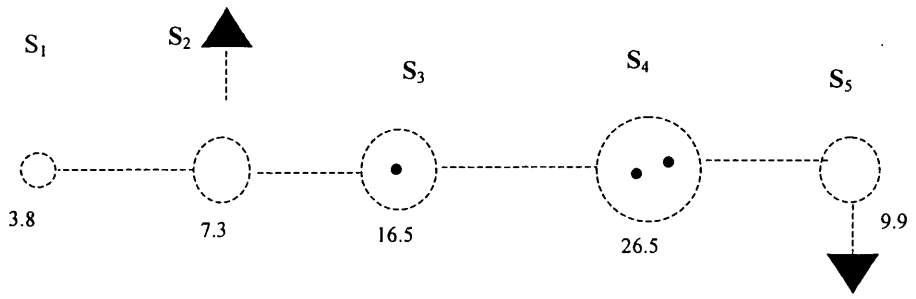


Figure 7. 17 Distributing copies with VWR - 2

Figs 7.16 and 7.17 illustrate the difference in reliabilities when the spares are distributed in two network topologies where the copies are identically distributed. When using the Voting With Regeneration technique, one must be aware of the effect on the reliability when distributing the spares in relation to the copies. The following table illustrates the differences in the two topologies displayed above.

S_1	S_2	S_3	S_4	S_5
3.3/3.8=0.9	6.1/7.3=0.8	14.6/16.5=0.9	150/26.5=5.7	13.9/9.9/=1.4

By comparing the topology in Fig 7.16 with the topology in Fig 7.17, for subnets S_1,..S_3, there is a small reduction in reliability when the spares are distributed. However, there is a significant increase in reliability for subnet S_4 and S_5.

So far, when we have observed the distribution of copies in a chain topology, we have clearly seen that keeping the copies together generates higher reliabilities. Here we notice that the distribution and design for the second topology yielded reliability results that were more balanced, whereas the first topology (Fig 7.16) yielded reliability results that were favourable to subnets S_4 and S_5. This means that when copies are distributed, VWR can also be used to enhance the reliability for topologies by creating a more balanced set of reliabilities as opposed to the negative exponential decrease in reliability as one moves away from the majority of copies.

7.3.4 SUMMARY OF RESULTS FOR THE DISTRIBUTION OF COPIES IN CHAIN TOPOLOGIES

We have confirmed that higher reliabilities are generated when copies are kept closer together. If copies are distributed, then higher reliabilities are generated when they are kept as close together as possible.

This work has made it possible to observe the effect on reliability when one distributes copies and witnesses. We have seen that the positioning of the witnesses in relation to the copies in a chain topology can affect reliability.

The positioning of spares in relation to copies when using the VWR technique also affects reliability. We have discovered that in addition to improving reliability compared to WV, the positioning of spares can be used in the design of networks to balance reliability from subnets when copies must be distributed.

7.4 RINGS

Ring topologies provide the opportunity to investigate the variation in reliability by the addition of links or the provision of more routes to the majority of copies.

7.4.1 COMPARATIVE RELIABILITY OF A NON-REPLICATED OBJECT

The addition of links intuitively improves the reliability of a topology since there is more than one way of accessing an object. Consider a network that consists of a non-replicated object where only the links fail and recover as in Fig 7.18.

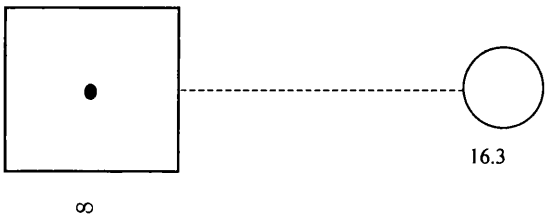


Figure 7. 18 topology with one link

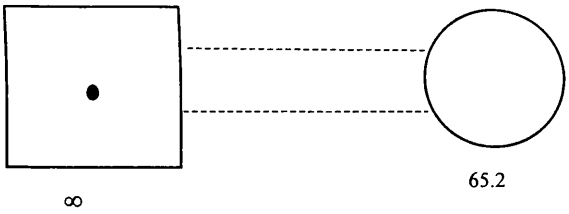


Figure 7. 19 topology with 2 links

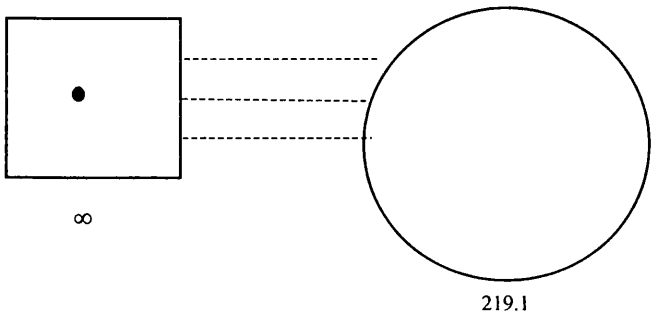


Figure 7. 20 The addition of links to a topology

The addition of one link to the network with a non-replicated object (figure 7.19) generates an increase in the reliability of a factor of 4, where as an additional two links (figure 7.20) improve the reliability by a factor of 13. As a reminder to the complexity of the theoretical calculation for Fig 7.19 please refer to 5.1.5.

7.4.2 THE ADDITION OF A LINK TO A CHAIN TOPOLOGY

The addition of a link to a chain with all copies in the same subnet can improve reliability even when both nodes and links fail. Consider the chain in figure 7.5 with an additional link linking subnet S_1 and S_5 as illustrated in figure 7.21.

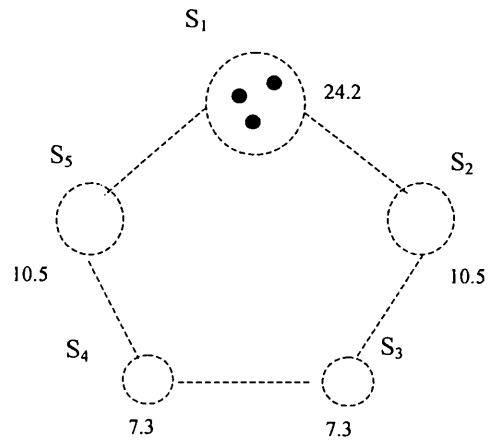


Figure 7. 21 Replication of object in rings

The area value for S_1 is unchanged since the replicas were placed in this subnet. The comparative reliability measure for S_2 is 1.3, which is an increase in reliability of 30%. For S_3 the increase in the reliability is 70% (comparative reliability 1.7). It is easy to see why the increase in reliability in S_3 is greater. The route to S_3 from the majority of copies in the chain (Figure 7.5) encompassed two links. With the additional link that made up the ring, there is another route from S_3 to the majority of copies of three links, $\{S_3,S_4,S_5\}$. The increase in reliability from S_2, with an additional link is not as significant since in the chain topology the probability of failure of the original route, of one link $\{S_2,S_1\}$ is lower than the probability of two links failing as in the case of S_3. Furthermore, the additional route for S_2 in case of failure must go through four links that may also fail.

It is also interesting to consider the addition of a link to a chain between two subnets, to combine the ring and chain topologies and examine the increase in reliability. For example, consider the topology in Fig 7.22 with an additional link between subnet S_1 and S_3 as illustrated by Fig 7.23.

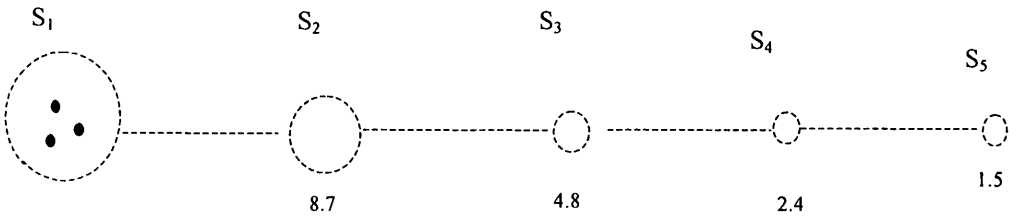


Figure 7. 22 Chain topology

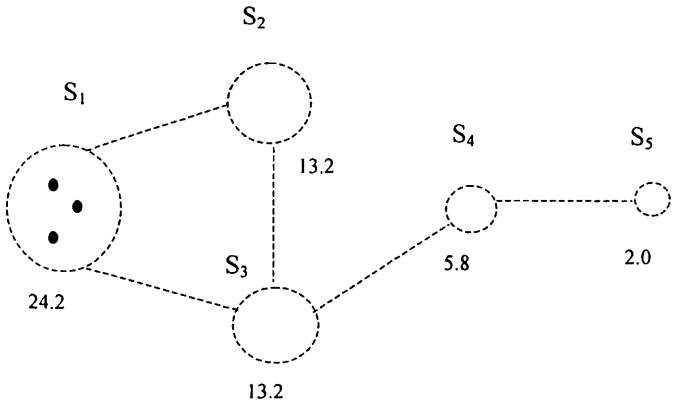


Figure 7. 23 Additional links in a chain topology

We can compare the reliability as follows:

S_1	S_2	S_3	S_4	S_5
24.2/24.2=1.0	13.2/8.7=1.5	13.2/4.8=2.8	5.8/2.4=2.4	2.0/1.0=2.0

The additional link does not affect the reliability from subnet S_1. The subnet view from S_2 has improved reliability since there is an additional route to the majority of copies. The subnet view from S_3 has an even greater increase in reliability since there is a faster route to the majority with the additional link. The rest of the subnet view is also improved simply because there is an additional link along the chain.

We can see therefore, that the addition of a link impacts the chain topology even when it is across some subnets and not others (indirect).

7.4.3 THE IMPACT OF ADDITIONAL LINKS ON THE CONSISTENCY CONTROL TECHNIQUE

The addition of links does not alter any of the observations regarding the chain topologies that were outlined above. The fact that there are two routes to the majority does not alter the fact that keeping replicas close together or on the same subnet generates higher reliabilities than distributing replicas. Similarly, in the case of VWW, higher reliabilities are attained, in any two identical ring topologies with similar copy placements, if the copies are kept together or are as close as possible to each other. Also in the case of Voting with Regeneration, higher reliabilities are produced when the spares are kept closer to the copies.

7.4.4 DISTRIBUTION OF COPIES IN A RING TOPOLOGY

The addition of links should help in curtailing reduced reliability when distributing copies. Consider the chain in Fig. 7.13 with an additional link between subnet S_1 and S_5, as illustrated by Fig. 7.24

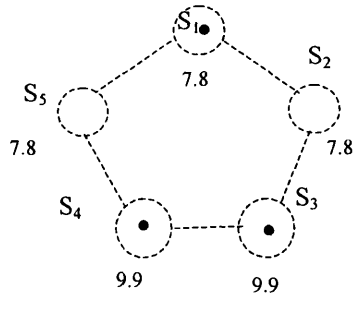


Figure 7. 24 Distributing copies in a ring

The comparative reliability (in relation to the chain topology displayed in Fig. 7.13) is presented in the table below:

S_1	S_2	S_3	S_4	S_5
7.8/3.1=2.5	7.8/3.9=2.0	7.8/5.3=1.5	9.9/3.9=2.5	9.9/3.1=3.2

The introduction of the additional link has generated a significant increase in the reliability for all the subnet views.

Discussion

Just as the introduction of spares can improve the reliability and provide greater balance when distributing copies, so can the addition of links. One can use the fact that additional links produce increases in the reliability to create greater balance in the reliabilities for network topologies, or designs. The comparative reliability measure can assist in investigating the impact of additional links (or redundancy) placed in strategic places in network topologies and designs.

7.5 STARS

Network configurations that are based on the star formation are very popular in industry; Ethernet and typical server/client type networks are based on this topology. Stars provide an opportunity to study situations that involve a subnet with several links, each connecting it to other subnets.

7.5.1 COMPARATIVE RELIABILITY OF A NON-REPLICATED OBJECT

A non-replicated object, placed in the center of the star, displays the same properties in a star as an object placed in a chain with one link (Fig. 7.25). This value can then be used as a reference to compare reliabilities.

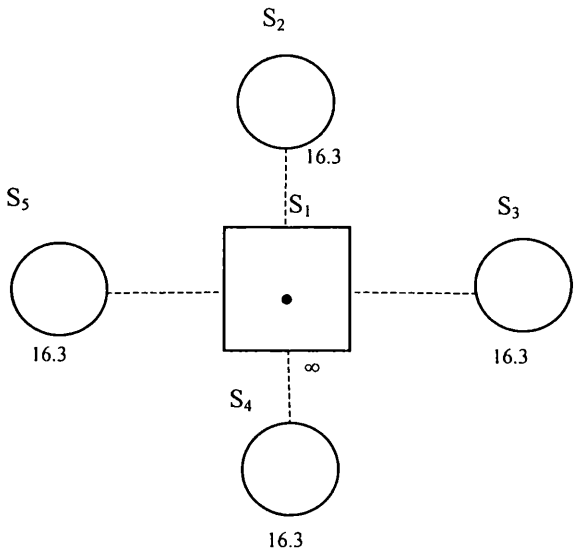


Figure 7. 25 Non-replicated object in a star

7.5.2 COMPARATIVE RELIABILITY OF A REPLICATED OBJECT

When placing the copies in the central subnet (S_1) the reliability is consistent with what has been observed for chain topologies with distances of one link (Fig. 7.26)

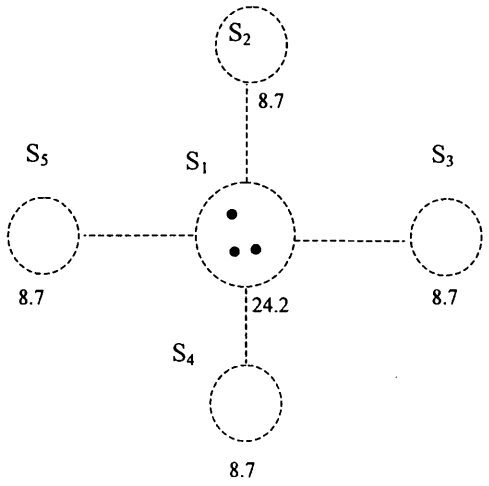


Figure 7. 26 Replicated object in a star - 1

Similarly, when the copies are in any other subnet (but the central one), the observations made above for chains across two subnets and their links apply as in Fig. 7.27.

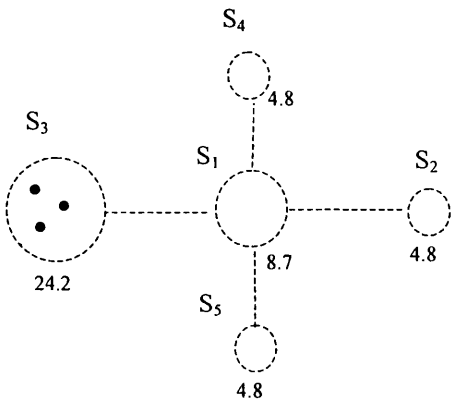
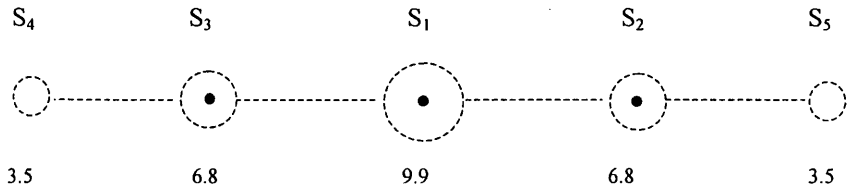


Figure 7. 27 Replicated object in a star - 2

The area value from subnet S_2 , for example, is equivalent to the a value of a subnet two links away from the majority of the copies in a chain topology (see Figure 7.5).

7.5.3 DISTRIBUTION OF COPIES IN A STAR

The distribution of copies in a star topology follows the distribution of copies in a chain very closely depending on where the copy placements are made and the distance from the majority of copies. For example, consider the chain and star in Fig. 7.28 with respect to an object across one link using WV.



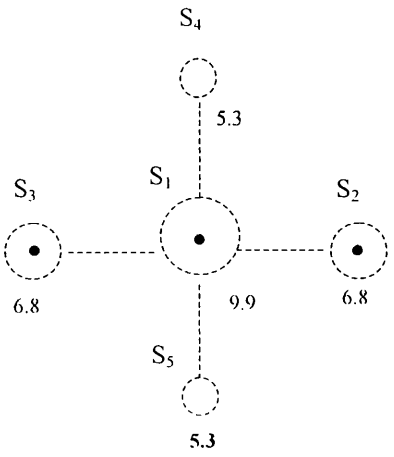


Figure 7. 28 Comparison of Distribution of copies in a star and chain

Comparative reliabilities are very similar for similar distances from the majority. However, the star topology still has the advantage since subnets S_4 and S_5 can collect a majority of copies faster in the star topology for similar copy distributions. However, if we distribute the copies in subnets that are not central, the reliabilities are reduced (figure 7.29) since we have increased the distance from the majority of the copies for some subnet views.

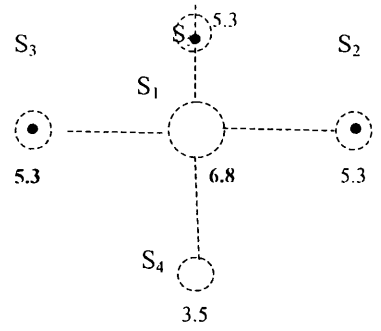


Figure 7. 29 Distribution of copies in a star - 1

It is easy to see that placing at least one copy in the central subnet would produce greater reliabilities, since it is faster to access at least one of the copies centrally. We are again reminded that keeping the copies together generates greater reliabilities provided the copies are in the central subnet. Compare Fig. 7.27 where the copies are kept together in a non-central subnet, against the distribution of the copies in Fig. 7.28,

where there is one copy in the central subnet. The relative reliabilities are shown in the table below:

S_1	S_2	S_3	S_4	S_5
9.9/8.7=1.1	6.8/4.8=1.4	6.8/24.2=0.3	5.3/4.8=1.1	5.3/4.8=1.1

We can see that the reliabilities are greater for every subnet view in the topology with the copies distributed, with the exception of the subnet containing all the copies.

Let us examine the reduction in reliability by distributing just one copy across the network as opposed to keeping all the copies together in the central subnet. We can compare Fig. 7.26 which has all the copies in the central subnet with the topology of Fig. 7.30 which has a copy placed in subnet S_3.

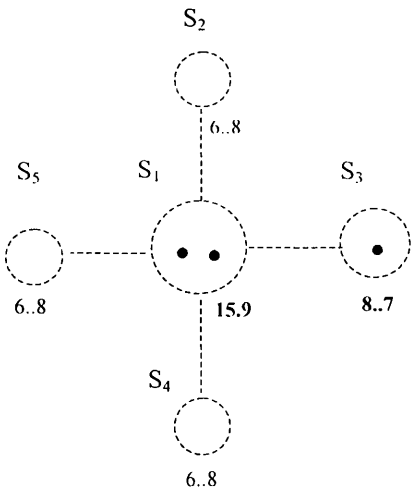


Figure 7. 30 Distribution of copies in a star - 2

The relative comparison of each subnet view is as follows:

S_2	S_3	S_1	S_4	S_5
8.7/6.8=1.3	8.7/8.7=1.0	24.2/15.9=1.5	8.7/6.8=1.3	8.7/6.8=1.3

This means that by distributing one of the copies to an adjacent subnet rather than keeping the copies together in the central subnet, you can reduce reliability by 34%.

Voting With Witnesses

Putting at least one "real" copy in the central subnet produces slightly higher reliabilities than putting only a witness in the central subnet. This is because there is an increased chance of getting a majority that includes at least one "real" copy when a copy is centrally located.

Voting With Regeneration

The distribution of the spare nodes can make a difference in comparative reliability. For example, consider the topologies in Fig. 7.31 and 7.32.

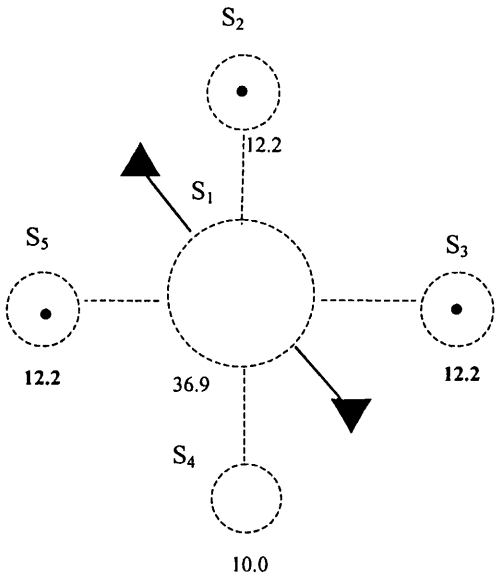


Figure 7. 31 Distribution of copies in a Star

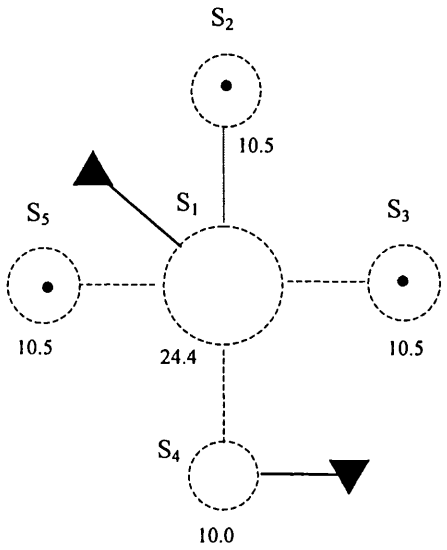


Figure 7. 32 Distribution of copies in a star

The copy placements in these topologies are equivalent, the only difference between them is the positioning of one spare node. In Fig. 7.31 both spare nodes are placed in the center, while in Fig. 7.32 one spare node is placed on a subnet not containing a copy and away from the center. The reliabilities in Fig. 7.31 are visibly greater.

It is obvious however, that when the copies and spares are placed in the central subnet and kept together, the reliabilities are higher.

7.6 DISCUSSION

The numerous results of these experiments are interesting and varied. The examples presented in this chapter were selected to delineate the various scenarios and circumstances that became apparent as a result of this study.

Previous research that included an analysis of file accessibility considers systems that do not allow partitioning failures. This assumption does not take into consideration the

topology of the network involved. The evidence from the results of this study show that when partitioning is assumed the reliability is significantly different.

For example, if it is assumed that a system does not partition the implementation of VWR with two spares generates a system that is nine times more reliable than the implementation of WV. However, it has been illustrated that once partitioning is introduced this increase is only applicable to a subnet that contains all the copies and the spares. The reliability from the other subnets is increased by a smaller factor when the copies are placed in the central subnet.

It is clearly seen from the analysis of the results that the network configuration has an impact on reliability. It has been illustrated that the addition of just one link between two subnets in a chain can increase the reliability by a factor of 2.

The illustrations above have corroborated that the placement of copies also affects reliability. We see that it is important to keep copies as close together as possible in topologies that consist of large rings or chains. If a more clustered topology is used then the copies should be distributed centrally or kept together centrally. We have furthermore shown that the addition of strategically placed links to a network topology can improved reliability when copies are distributed.

If balancing the cost of storage against performance is necessary, it has been shown that using the VWW technique is preferable since the reduction in reliability is small due to a copy being replaced by a witness, especially when replicas are kept together. We have also been able to show that the positioning of witnesses in a topology affects reliability. Using the comparative reliability measure, one is able to quantify the reduction in reliability when using witnesses. We are also able to compare different placements of copies and witnesses to determine where would be the most efficient placement of replicas.

It has been illustrated that VWR can provide a considerable increase in reliability compared to WV if spare nodes are placed in the same subnets as at least one of the copies. Moreover, the fact that regeneration improves reliability can be used in conjunction with the comparative reliability measure to assist computer professionals in

determining where to position spares in order to improve the reliability of network topologies, especially when copies are distributed.

Chapter 8
Conclusions and Further Work

This chapter reviews the objectives of the dissertation and demonstrates how each objective has been accomplished. A summary of original work is given followed by a discussion of further work.

8.1 SUMMARY OF THESIS

The thesis of this dissertation is that using comparative reliability is a satisfactory and practical technique for the comparison of replication control algorithms. Four objectives were established in the introduction to demonstrate the thesis:

- Find a comparative reliability measure that captures all the significant information of the reliability function in one value.

- Determine a suitable notation technique that represents clearly and concisely all the numerous parameters and results that are involved in a comparative study of different network configurations, control algorithms and copy placements.

- Find a practical technique for a comprehensive investigation of the resiliency of replication algorithms to changes in network configuration and copy placement.

- Combine the work on reliability measures, notation techniques and the results of the investigation to draw constructive conclusions about the impact of the network configuration and copy placements on the reliability of Distributed Systems.

The subsections below describe how each of these objectives was accomplished.

8.1.1 FINDING A COMPARATIVE RELIABILITY MEASURE

In Chapter 2, a distributed system was defined as a set of subnets that contain a fully connected set of nodes joined together by links. Because the system can partition, the view from one subnet may be different to that of another subnet. Using the conventional reliability function to analyze a network configuration by comparing reliabilities from several subnet views can be very confusing and tedious. The comparison of different systems that allow partitioning can be even worse.

Suitable comparative measures were investigated with the aim of maintaining as much of the information of the reliability function as possible in one value. This value can then be used in an intuitive manner, rather like availability, to determine whether reliability is good by comparing the value with one of another reliability function or a relative reference value.

The reliability function is manipulated to derive three different measures of comparative reliability: the decay factor, maximum difference and area ratios. These measures were outlined in Chapter 4.

Decay Factor

The Decay Factor is the gradient of the reliability function. It can be used to highlight the differences in the rate of decay from different subnet views. A disadvantage with this comparative measure is that the assumption that the reliability function is a negative exponential may be too stringent since, reliability curves are "flattened" near the y-axis and vital information may be ignored or lost. After some study this measure was abandoned, since it had no intuitive value, and because in order to achieve a "best fit" to the experimental reliability curve, weighted regression had to be used.

Maximum Difference

The Maximum Difference compares two reliability functions by calculating their ratio at the point of maximum difference. Problems occurred with this comparative reliability measure when comparing several reliability functions that could lead to misleading interpretations about the reliabilities.

Area Ratios

The Area Ratios involve the comparison of the value of the area of a reliability function with one view, with reference to another. This measure was chosen as the measure for this investigation since it has an intuitive value and it offers a great strategy for understanding the comparative reliabilities of different network topologies and placement of copies. Furthermore, it allows reliability of network topologies that can partition to be compared against the reliability of networks that do not.

8.1.2 DETERMINING A SUITABLE NOTATION

In a presentation of an analysis of an investigation, it is important to inform the observer of all variables that have affected the results. For an investigation that entails the resiliency of replication to changes in network topology and copy placement one has to describe the network topology accurately, i.e. the position of subnets and links, the placement of copies, whether the nodes and links fail and recover or whether they are perfect. In the case of regeneration, the positioning of spares needs to be displayed.

A presentation technique that visualizes clearly all these factors was developed in Chapter 4. It was used to illustrate the results of the investigation using area ratios. This notation technique represents visually:

- the topology of the network in terms of the position of the nodes and links of a network configuration,
- whether the nodes and links fail and recover,
- the placement of copies within that topology,
- the relationship of the reliabilities of different subnet views,
- attributes of the algorithm present, such as which of the replicas are witnesses (in the case of VWW) and where the regenerated copies are (in the case of VWR).

Hence, in one glance one is able to discern whether the topology is able to partition, whether one subnet view is better than another, whether one copy placement is better

than another, or whether one algorithm is more reliable than another with equivalent network configuration!

This technique proved to be remarkably successful in presenting the results in Chapter 7. Each configuration is easy to digest and allows the reader to understand the numerous parameters involved without effort. This means that the reader is able to concentrate on the most important part of the exercise, which is the analysis of reliability for each configuration.

8.1.3 INVESTIGATION TECHNIQUE

Chapter 5 rationalizes why simulation was chosen as the approach for the collection of reliability information. Concatenated Local-area and Wide-area Network (CLOWN) described in Chapter 6, was chosen as the simulation environment with which to experiment and investigate different topologies and replication techniques. CLOWN was very successful in managing the vast number of experiments that had to be executed. Once the program for each of the consistency control technique chosen was written, it was easy to construct each topology. The fact that one is able to view the network configuration while it was being executed was very convenient, especially since there are many permutations of the same topology with different copy placements, witnesses or spare nodes.

Three popular consistency control techniques namely, weighted voting, voting with witnesses and voting with regeneration were implemented, since they can withstand partitioning and a small number of replicas.

Experiments were performed to investigate the resiliency of these techniques to changes in the network configuration and the placement of copies.

Network topologies were classified into three basic categories: chains, rings and stars, since any network topology can be constructed from a combination of any of these. Using these categories an exhaustive study of all combinations of five subnets and three copies was conducted. This combination was chosen because the interest in replication

for a small number of replicas has increased and in the interest of the time to complete this work.

The work on reliability measures, notation techniques and the results of the investigation was combined to draw constructive conclusions about the impact of the network configuration and copy placements on the reliability of Distributed Systems.

8.1.4 ANALYSIS OF RESULTS

The area ratios were used to compare and analyze the impact of the network topology on the reliability without the effect of the replication algorithm and without partitioning. This was followed by the analysis of the reliability for the view of the network from every subnet.

Not only can one compare the reliabilities of different views intuitively using the comparative measure, the notation technique is also used to present the results visually. This means that the reader is able to discern whether the network can partition, where the copies (witnesses and replicas) are, and whether one subnet view is better than another.

Using the classification of the network topologies into chains, rings and stars, an understanding of the behaviour of each of these different topologies was seen.

One is also able to evaluate how the reliability of one topology with a particular algorithm is compared with the reliability of another. Hence we are able to study the reliability of similar topologies with different algorithms and compare them.

8.2 IMPACT OF NETWORK TOPOLOGY AND COPY PLACEMENT ON RELIABILITY

An inspection of network configurations led to the conviction that any arbitrary network configuration can be constructed from any of the following basic network topologies: chains, rings and stars. This classification provided the basis of the investigation of the resiliency of Weighed Voting, Voting With Witnesses and Voting With Regeneration to changes in the network topology.

In Chapter 7 the results of topologies for the case where there are no partitioning failures were analyzed first. Once partitioning is introduced, the comparative reliability was considered from each subnet view in the topology. For each topology all possible copy placements including witnesses and regenerated copies were investigated and compared both against other configurations for the same consistency control and for similar topological configuration but different consistency control techniques.

The results can be summarized as follows:

1. When partitioning is assumed, reliability is significantly different.

2. The decrease in reliability as the distance in the number of links from the majority of copies increases follows some kind of negative exponential function.

3. The decrease in reliability in using less storage with VWW compared with WV is estimated at 7%.

4. The addition of a link to a chain topology between any two subnets produced a significant increase in reliability for all the subnet views and different copy placements.

5. It is important in topologies that consist of large rings or chains to keep copies close together. If a more clustered topology is used then the copies should be distributed centrally or kept together centrally.

6. Regeneration can provide a considerable increase in reliability compared to WV if spare nodes are placed in the same subnets as at least one of the copies.

8.3 CONCLUSION

Many of the observations that were derived from the results of the simulation experiments are satisfyingly, just as one would have expected.

The fact that the reliability is significantly different when partitioning is introduced can be rationalized. However, this thesis has facilitated the quantification of the significance of partitioning, of how much less reliable a topology is when partitioning is introduced.

We are able to visualise the fact that the decrease in reliability, as the distance in the number of links from the majority of copies increases, follows some kind of negative exponential function.

In the case of the Voting With Witness technique, which offers the advantage of less storage space, we are able to quantify the decrease in the reliability for this technique compared with Weighted Voting. This information is very valuable when a decision to implement one algorithm as opposed to another depends on high cost constraints for storage.

For performance issues, using the techniques outlined in this thesis we are able to quantify the increase in reliability due to one additional link in a chain topology, regardless of the copy placement or subnet view.

This thesis has provided an exciting approach to understanding the importance of keeping replicas as close together as possible in any type of topology. Evidently, there are many reasons not to keep copies together, which means that one has to find a balance between other constraints and performance. The most reliable replicated object would be one replicated several times on a single perfect server. In the absence of perfect servers, moving objects onto other nodes removes the single point of failure, but introduces others; the links between the nodes. One of the main achievements is to

have a set of results and a data method of data presentation that can be used to assess the effects of not keeping replicas close together.

This work has offered a means by which one can investigate how best to distribute replicas in distributed systems.

One is able to compare consistency control techniques before implementation in order to investigate which one would suite their environment, in a realistic manner. For example, two popular consistency control techniques that have been chosen for this work allow administrators to quantify how much more reliable their network configuration is when Voting With Regeneration is implemented as compared to Weighted Voting. All this without disrupting the normal day-to-day activities of their organisation.

The fact that we have built an understanding of the basic components of networks has equipped us to make intuitive inferences about the reliability of any network configuration that is employing replication.

The techniques outlined in this thesis have provided an intuitive comparative measure and an agreeable notation technique for the investigation and presentation of the performance and impact of network configurations on the reliability of Distributed Systems.

8.4 FURTHER WORK

The measure of comparative reliability provides a tool for further investigation of the resiliency of replication algorithms to changes in the network topology and copy placement. Larger numbers of subnets and more copies could be considered. If the permutations of all the copy placements for each type of network are too large a task, one can use the findings in this work as a guide to the valuable configurations to implement and study. Also different replication algorithms could be analysed and compared with each other.

This work has facilitated the analysis of a large number of variables at a glance. Since the author believes that Optimistic Consistency Control Techniques will play an important role in the future of replication, one can expand the techniques developed in this work to study the resiliency of optimistic replication techniques and the impact of network configuration on the reliability when they are implemented.

The results of the simulation experiments showed that partitioning significantly affects reliability. However, there is little data on how likely partitioning is, and whether it can be prevented to a significant degree. It is important to understand and quantify partitioning.

Comparative reliability measures provide the opportunity to examine a particular network topology and highlight all the best placement of copies. This idea could be developed as a performance tool for system administrators to examine their own network. Such a program could also highlight the reliability from different subnet views according to different copy placement. Experimentation on more Concurrency Control Algorithms can be executed to provide a comprehensive guide on performance issues relating to the different techniques.

The thesis can assist applications such as video conferencing, where continuous period of access are important to each video source or some central reflector.

References

1. J.A. Abraham
   "An Improved Algorithm for Network Reliability"
   IEEE Transactions On Reliability, Vol R-28, No 1, Apr 1997

2. *D. Agrawal and El Abbadi*
   "Integrating Security with Fault-tolerant Distributed Databases"
   The Computer Journal, Vol 33, No 1, Feb 1990

3. P.A. Alserg and J.D. Day
   "A principle for resilient sharing of distributed resources"
   In Proceedings, 2nd International Conference on Software Engineering, IEEE Computer Society Press, Oct 1976, pp 562-p570

4. G.R. Andrews and R.A. Olsson
   "Report on the SR Programming Language - Version 1.1
   Department of Computer Science, The University of' Arizona, Tucson, Arizona, May 1989

5. B.S Bacarisse and S. Bek Baydere
   "A Low Cost File Replication Algorithm"
   Proceedings of the 34th Annual IEEE International Conference, COMPCON Spring 89, San Francisco, CA, Feb 1989

6. B.S Bacarisse and S. Bek Baydere
   "Reliability of Replicated Files in Partitioned Networks"
   University College London, 1989

7. B. Bacarisse and S. Bek-Baydere
   "A Low Cost File Replication Algorithm"
   IEEE COMPCON Spring'89 conference in San Fransisco, Feb 1989

8. D. Barbara and H. Garcia-Molina
   "Evaluating Vote Assignments With A Probabilistic Metric"
   June 1985

9. D. Barbara, H. Garcia-Molina and A. Spauster
"Increasing Availability Under Mutual Exclusion Constraints with Dynamic Vote Reassignment"

10. R.E. Barlow and F. Proschan
"Mathematical Theory of Reliability"
Wiley, New York, 1965

11. R.E. Barlow and K.D. Heidtmann
"Computing k-out-of-n System Reliability"
IEEE Transactions on Reliability, Vol R-33, No. 4, pp.322-323, Oct 1984

12. S. Baydere
"A Practical Consistency Scheme for File Replication"
PhD thesis, University College London, July 1990

13. S. Baydere and B. Bacarisse
"Reliability of Replicated Files in Partitioned Networks"
IEEE Workshop on Management of Replicated Data, Houston, Nov 1990

14. P.A. Bernstein and N. Goodman
"Concurrency Control in Distributed Database Systems"
ACM Computing Surveys, Vol 13, No 2, June 1981, p185-p221

15. P.A. Bernstein and N. Goodman
"The failure and recovery problem for replicated databases"
Proceedings of the 2nd Annual Symposium on Principles of Distributed Computing, ACM, 1983, p114-p122

16. P.A. Bernstein and N. Goodman
"An Algorithm For Concurrency Control And Recovery In Replicated Databases"
ACM Transactions on Database Systems, Vol 9, No 4, Dec 1984, p596-p615.

17. K.P. Birman, T.A. Joseph, T. Raeuchle, and A. El Abbadi
"Implementing fault-tolerant distributed objects"
In Proceeding 4th Symposium on Reliability in dstributed Software and Database Systems, Oct 1984

18. K.P. Birman
"Replication and fault-tolerance in the ISIS system"
In Proceeding of the 10th Symposium on Operating Systems Principles, Dec 1985, p79-p86

19. J.J. Bloch, D.S. Daniels and A.Z. Spector
"A Weighted Voting Algorithm for Replicated Directories"
Journal of the ACM October 1987

20. Luis-Felipe Cabrera and Jehan-François Pâris
"Replicated Data"
Proceedings of the Workshop on Management of Replicated Data, IEEE Computer Society Press, Houston, Texas, November 8-9, 1990.

21. Chun-Kin-Chan; Doumi-T; Tortorella-M
"Power-related failure mechanisms in the analysis of wireless system availability"
Annual Reliability and Maintainability Symposium. 2000 Proceedings. International Symposium on Product Quality and Integrity (Cat. No.00CH37055). IEEE, Piscataway, NJ, USA; 2000; xviii+394 pp. p.335-40.

22. Chan, S. Fox, W.T. Lin, A. Nori, and D. Ries
"The implementation of an integrated concurrency control and recovery scheme"
Proceedings of the 1982 SIGMOD Conference: International Conference on Management of Data 1982, p11-p24

23. Chan and D. Skeen
"The Reliability Subsystem of a Distributed Database Manager"
Technical Report CCA-85-02, Computer Corporation of America, 1985

24. D.J. Chen and T.H. Huang
"Reliability Analysis of Distributed Systems Based on a Fast Reliability Algrorithm"
IEEE Transactions on Parallel and Distributed Systems, Vol 3, No 2, Mar 1992

25. D.T. Chiang and Shun-Chen Nui
"Reliability of Consecutive-k-out-n:F System"
IEEE Transactions on Reliability, Vol R-30, No. 1, pp.87-89,April 1981

26. D. Davcev and W.A. Burkhard
"Consistency and Recovery Control for Replicated Files"
Proceedings of the 10th ACM Symposium on Operating System Principles 1985 p87-p96

27. S.B. Davidson,
"Optimisim and consistency in partioned distributed database systems"
ACM Transactions on Database Systems, Vol 9, No 3, p456-p482, Sep 1984

28. S.B. Davidson, H. Garcia-Molina and D. Skeen
"Consistency in Partitioned Networks"
ACM Computing Surveys, Vol 17, No 3, September 1985

29. D.L. Eeger and, K.C. Sevcik
"Achieving Robustness in Distributed Database Systems"
ACM Transactions on Database Systems, Vol 8, No 3, p354-p381, Sept 1983

30. El Abbadi, D. Skeen, and F. Christian
"An Efficient, Fault Tolerant Protocol for Replicated Data Managment"
Proceedings of the 4th ACM SIGACT-SIGMOD Symposium on Principles of Database Systems, Mar 1985, p215-p229

31. El Abbadi and S. Toueg
"Maintaining Availability in Partitioned Replicated Databases"
ACM Transactions on Database Sys, vol 14, No 2, June 1989

32. El-Koliel-MS; Borysiewicz-M
"Computer program for the Markovian reliability and availability as applied to emergency AC power system configurations of the nuclear power plants"
Nukleonika. vol.44, no.3; 1999; p.439-56.

33. K.P. Eswaran, J.N. Gray, R.A. Lorie and I.L. Traiger
"The Notions of consistency and predicate locks in a database system"
Communications of the ACM, Vol 19, No 11, Nov 1976, p624-p633

34. M. Fischer and A. Michael
"Sacrificing serializability to attain high availability of data in an unreliable network"
In Proceeding, ACM SIGACT-SIGMOD Symp. on Principles of Database Systems, Mar 1982, p7-p75

35. L. Fratta and G. Montanari
"A Recursive Methods Based on Case Analysis for Computing Network Terminal Reliability"
IEEE Transactions on Communications, Vol COM-26, No 8, Aug 1976

36. H.Garcia-Molina
"Election in a Distributed Computing System"
IEEE Transactions on Computers, Vol 3, No 1, Jan 1982, p48-p59

37. H.Garcia-Molina,T.Allen,B.Blaustein,R.M.Chilenskas,D. Ries
"Data-Patch: Integrating Inconsistent Copies of a Database after a Partition"
In Proceedings of the 3rd Symposium on Reliability in Distributed Software and Database Systems, Oct 1983

38. D.K. Gifford
"Weighted Voting for Replicated Data"
ACM 1979

39. H.M. Gladney
"Data Replicas in Distributed Information Services"
ACM Transactions on Database Systems, Vol 14, No 1, March 1989

40. B.V. Gnedenko, Yu.K. Belyayev and A.D. Solovyev
"Mathematical Methods of Reliability Theory"
Academic Press Inc. 1969

41. N.Goodman, D.Skeen, A.Chan, U.Dayal, S.Fox and D.Ries
"A recovery algorithm for a distributed database system"
In Proceedings, 2nd ACM SIGACT-SIGMOD Symp. on Principles of Database Systems, Mar 1983

42. Golding-R; Borowsky-E
"Fault-tolerant replication management in large-scale distributed storage systems"
Proceedings of the 18th IEEE Symposium on Reliable Distributed Systems. IEEE Comput. Soc, Los Alamitos, CA, USA; 1999; xii+402 pp. p.144-55.

43. Golick-J
"Distributed data replication"
Network-Magazine. vol.14, no.12; Dec. 1999; p.60-2, 64.

44. H. Gupta and J. Sharma
"A Method of Symbolic Steady-State Availability Evaluation of k-out-of-n:G System"
IEEE Transactions on Reliability, Vol R-28, No. 1, pp.56-57, April 1979

45. T. Haerder, and A. Reuter
"Principles of Transaction-Oriented Database Recovery"
ACM Comupring Surveys, Vol 15, No 4, p287-p317, Dec 1983

46. M.M. Hammer, and D.W. Shipman
"Reliability Mechanisms in SDD-1, a System for Distributed Databases."
ACM Transactions on Database Systems, Vol 5, No 4, p431-p466, Dec 1980

47. Han-Yang-Cheng; Chung-Ta-King
"File replication for enhancing the availability of parallel I/O systems on clusters"
IEEE Computer Society International Workshop on Cluster Computing. IEEE Comput. Soc, Los Alamitos, CA, USA; 1999; xvii+358 pp. p.137-44.

48. S. Hariri and C.S. Raghavendra
"SYREL: A Symbolic Reliability Algorithm Based on Path and Cutset Methods"
IEEE Transactions on Computers, Vol C-36, No 10, October 1987

49. S. Hariri, C.S. Raghavendra and V.K. Prasanna Kumar
"Reliability Analysis in Distributed Systems"
Proceedings of the 9th International Conference on Distributed Computing Systems, June 1989

50. M. Herlihy
"Using Type Information to Enhance the Availability of Partitioned Data"
Sep 1985

51. M. Herlihy
"A Quorum-Consensus Replication Method for Abstract Data Types"
ACM Transactions Feb 1986

52. M. Herlihy and J.D. Tygar
"How to Make Replicated Data Secure"
Advances in Cryptology, Proceedings of CRYPTO'87, Springer Verlag, Lecture Notes in Computer Science, No 293, pp 379-391, 1987

53. M. Herlihy
"Optimistic Concurrency Control for Abstract Data Types"
Operating Systems Review, Vol 21, No 2, April 1987

54. M. Herlihy
"Apologizing Versus Asking Permission: Optimistic Concurrency Control for Abstract Data Types"
ACM Transactions on Database Systems, Vol 15, No 1, March 1990

55. M. Herlihy
"Concurrency and Availability as Dual Properties of Replicated Atomic Data"
Journal of the ACM, Vol 37, No 2, April 1990

56. A.H. Hevesh
"Comments on : Steady-state Availability of k-out-of-n:G System with Single Repair"

IEEE Transactions on Reliability, Vol R-33, No. 4, pp.324, Oct 1984

57. Holliday-J; Agrawal-D; El-Abbadi-A
"Database replication: if you must be lazy, be consistent"
Proceedings of the 18th IEEE Symposium on Reliable Distributed Systems. IEEE Comput. Soc, Los Alamitos, CA, USA; 1999; xii+402 pp. p.304-5.

58. Hongzhou-Wang; Hoang-Pham
"Survey of reliability and availability evaluation of complex networks using Monte Carlo techniques"
Microelectronics-and-Reliability. vol.37, no.2; Feb. 1997; p.187-209.

59. Huseyin, G. Pavlou, and P.T. Kerstein
"A Distributed Database Study"
Technical Report 143, University College London, Computer Scienc Department, March 1988

60. R. Jain
"The Art of Computer Systems Performance Analysis"
John Wiley & Sons, Inc. 1991

61. S. Jajodia and D. Mutchler
"Dynamic Voting Algorithm for Maintaining the Consistency of a Replicated Database"
ACM Transactions on Database Systems, Vol 15, No 2, June 1990

62. A.M. Johnsons, JR. and M. Malek
"Survey of Software Tools for Evaluating Reliability Availability, and Serviceability"
ACM Computing Surveys, Vol 20, No 4, Dec 1988

63. T.A. Joseph and K.P. Birman
"Low Cost Management of Replicated Data in Fault-Tolerant Distributed Systems"
ACM Transactions of Computing Systems, Vol 4, No 1, Feb 1986

64. Kettinger-WJ; Lee-CC
"Replication of measures in information systems research: the case of IS SERVQUAL"
Decision-Sciences. vol.30, no.3; Summer 1999; p.893-9.

65. R.L. Keynon and R.J. Newell
"Steady-State Availability of k-out-of-n:G System with Single Repair"
IEEE Transactions on Reliability, Vol R-32, No. 2, pp.188-189, June 1983

66. C. Koelbel, G. Spafford, and G. Leach
"Workshop on Experiences with Building Distributed and Multiprocessor Systems"
ACM Operating Systems Review, Vol 24, No 2, April 1990

67. H.F. Korth
"Locking primitives in a database system"
Journal of the ACM, Vol 30, No 1, Jan 1983, p55-p79

68. P.A. Kullstam
"Availability, MTBF and MTTR for Repairable M out of N System"
IEEE Transactions on Reliability, Vol R-30, No. 4, pp.393-394, Oct 1981

69. L. Lamport
"Time, Clocks. and the Ordering of Events in a Distributed System"
Communications of the ACM, Vol 21, No 7, pp 631-653, Oct 1979

70. L. Lamport, R. Shostak and M. Pease,
"The Byzantine Generals Problem"
ACM Transactions on Computer Systems, Vol 4, No 3, pp 382-401, Jul 1982

71. Lemaire-M
"Reliability and mechanical design"
Reliability-Engineering-&-System-Safety. vol.55, no.2; Feb. 1997; p.163-70.

72. E. Levy and A. Silberschatz
"Distributed File Systems: Concepts"
ACM Computing Surveys, Vol 22, No 4, Dec 1990

73. W. Lobianco
"Survey on Software Fault-Tolerance"
Research Document RD-1, UCL, Aug 1990

74. Loheac-J-L; Raoult-F; Bonnaud-O; Taurin-M
"Analysis for the reliability of the intrinsic base ion implantation of a 3 GHz I/sup 2/L bipolar process from the measure of integrated resistances: From the results, setting of rules for an expert system"
Microelectronics-and-Reliability. vol.37, no.1; Jan. 1997; p.179-86.

75. D.E. Long and J.L. Carroll
"The Reliability of Regeneration-Based Replica Control Protocols"
9th International Conference on Distributed Computing Systems, Jun 1989

76. D.E. Long and J.L. Carroll
"Estimating the Reliability of Hosts Using the Internet"

77. D.E. Long
"Voting with Regenerable Volatile Witnesses"
July 1990

78. T. Mann, A. Hisgen, and G. Swart
"An Algorithm for Data Replication"
Technical Report:46, Digital Systems Research Center, June 1989

79. McDonald-AB; Znati-T
"A path availability model for wireless ad-hoc networks"
IEEE Wireless Communications and Networking Conference (Cat. No.99TH8466). IEEE, Piscataway, NJ, USA; 1999; 3 vol (xxviii+1580) pp. p.35-40 vol.1.

80. Mitrani
"Simulation techniques for discrete event systems"
Cambridge Computer Science Texts 14, Cambridge University Press, 1982

81. C. Mohan
"Distributed database management: Some thoughts and analysis"
Proceedings of the ACM Annual Conference, Oct 1980, p399-p410999

82. J.D. Musa
"The Measurement and Management of Software Reliability"
Proceedings of IEEE, 68, 1980, p1131-p1143

83. R. Needham and M. Burrows
"Locks in Distributed Systems - Observations"
Operating Systems Review, Vol 22, No 3, July 1988

84. J.D. Noe and Agnes Andreassian
Effectiveness of Replication in Distributed Computer Networks
7th International Conference on Distributed Computing Systems, Sept 1987

85. B.M. Oki and B.H. Liskov
"Viewstamped Replication: A New Primary Copy Method to Support Highly-Available Distributed Systems"
Proceedings of the Seventh Annual ACM Symposium on Principles of Distributed Computing, August 1988.

86. C.H. Papadimitriou and P. Kanellakis
"On Concurrency control by multiple versions"
ACM Transactions on Database Systems, Vol 9, No 1, Mar 1984 p89-p99

87. Pâris J.F.
"Voting with Witnesses - A Consistency Scheme for Replicated Files"
6th International Conference on Distributed Computing Systems 1986

88. G.J.Popek, B.Walker, J.Chow, D.Edwards, C.Kline, G.Rudisin, and G.Thiel
"LOCUS: A Network transparent, high reliability distributed system"
Proceedings 8th ACM Symposium on Operating Systems Principles, Dec 1981, p169-p177

89. Poulakidas-AS; Singh-AK
"Online replication of shared variables"
Proceedings of the 17th International Conference on Distributed Computing Systems (Cat. No.97CB36053). IEEE Comput. Soc. Press, Los Alamitos, CA, USA; 1997; xvii+596 pp. p.500-7.

90. Power J.
"Distributed System Evolution - Some Observations"
ACM Operating Systems Review, Vol 23, No 4, Oct 1989

91. C. Pu, J.D. Noe, and A. Proudfoot
"Regeneration of Replicated Objects: A Technique for Increased Availability"
April 1985

92. M.O.Rabin
"Efficient-Dispersal of Information for Security, Load Balancing, and Fault Tolerance"
Journal of the ACM, Vol 36, No 2, April 1989

93. C.S. Ragavendra, V.K. Prasanna Kumar and S.Hariri
"Reliability Analysis in Distributed Systems"
IEEE Transactions on Computers, Vol 37, No 3, Mar 1988

94. S. Rai and K.K. Aggarwal
"An Efficient Method for Reliability Evaluation of a General Network"
IEEE Transations on Reliability, Vol R-27, No 3, Aug 1978

95. D.P. Reed
"Implementing atomic actions on decentralized data"
ACM Transactions on Computer Systems, Vol 1, No 1, Feb 1983, p3-p23

96. R. van Renesse and A.S. Tanenbaum
"Voting With Ghosts"
Computer Society Press, June 1988

97. J.B. Rothnie and N. Goodman
"A survey of research and development in distributed database management"
Proceedings of the 3rd International Conference on Very Large Data Bases (VLDB), Oct 1977, p48-p61

98. M. Satyanarayanan
"Scalable, Secure, and Highly Available Distributed File Access"
Computer, May 1990

99. R.D. Schlichting and F.B. Schneider
"Fail-Stop processors: An approach to designing fault-tolerant computing systems".
ACM Transactions on Computing Systems, Vol 1. No 3, Aug 1983, 222-238.

100. T.J. Schriber
"An Introduction to Simulation Using GPSS/H"
John Wiley & Sons 1991

101. P.M. Schwarz and A.Z. Spector
"Synchronizing shared abstract types"
ACM Transactions on Computing Systems, Vol 2, No 3, Aug 1984, p223-p250

102. Siegel, K. Birman, and K. Marzullo
"Deceit: A flexible Distributed File System"
Paper by the Department of Computer Science, Cornell University, Ithaca, NY, Noverber 1989.

103. D. Skeen, F. Christian, A. El-Abbadi
"An Efficient, fault-tolerant protocol for replicated data management"
In Proceeding of the 4th ACM SIGACT-SIGMOD Conference on Principles of Database Systems, Mar 1985, p215-p229

104. Sloman M. (Editor)
"Network and distributed Systems Management"
Adisson-Wesley Publishing Company 1993

105. Sloman M. and Kramer J.
"Distributed Systems and Computer Networks"
Prenitce Hall 1987

106. Soh S. and Rai S.
"CAREL: Computer Aided Reliability Evaluator for Distributed Computing Networks"
IEEE Transactions on Parallel and Distributed Systems, Vol 2, No 2, Apr 1991

107. Sorensen, Soren-Askel
"Simulation and Control"
Proceedings of International Conference of Modeling, Bao Yuanlu Slun Lian, Hefei

Anluei, China Feb 1993

108. Sorensen, Soren-Askel and Jones M.G.W.
"The CLOWN Network Simulator"
Proceedings of the 7th UK Performance Engineering Workshop, Berlin,1991 123-130

109. Stonebraker M.
"Concurrency Control and Consistency of Multiple Copies of Data in Distributed INGRES"
IEEE Transactions on Software Engineering, SE-9, 3, May 1983, p219-p228

110. Sun-G; Mori-K
"Flexible and autonomous service replication technique"
IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No.99CH37028). IEEE, Piscataway, NJ, USA; 1999; 6 vol. (1179+1075+1106+1124+1140+1078) pp. p.113-18 vol.3.

111. Thomas R.H.
"A solution to the concurrency control problem for multiple copy databases"
In Proc. 16th IEEE Computer Society Internationl Conference (COMPCON), spring 1978

112. Thomas R.H.
"Consensus approach to concurrency control for multiple copy databases"
ACM Transactions on Database Systems, Vol 4, No 2, June 1979, p180-p209

113. Wendai-Wang; Kececioglu-DB
"Confidence limits on the inherent availability of equipment"
Annual Reliability and Maintainability Symposium. 2000 Proceedings. International Symposium on Product Quality and Integrity (Cat. No.00CH37055). IEEE, Piscataway, NJ, USA; 2000; viii+394 pp. p.162-8.

114. Wolf-T; Strohmeier-A
"Fault tolerance by transparent replication for distributed Ada 95"
Reliable Software Technologies - Ada-Europe '99. 1999 Ada-Europe International Conference on Reliable Software Technologies. Proceedings. (Lecture Notes in Computer Science Vol.1622). Springer-Verlag, Berlin, Germany; 1999; xiii+449 pp. p.412-24.