Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

FLORIAN SCHINTKE, ALEXANDER REINEFELD

On the Cost of Reliability in Large Data Grids

ZIB-Report 02-52 (December 2002)

On the Cost of Reliability in Large Data Grids

Florian Schintke, Alexander Reinefeld

Zuse Institute Berlin
{schintke,ar}@zib.de

Abstract. Global grid environments do not only provide massive aggregated computing power but also an unprecedented amount of distributed storage space. Unfortunately, dynamic changes caused by component failures, local decisions, and irregular data updates make it difficult to efficiently use this capacity.
In this paper, we address the problem of improving data availability in the presence of unreliable components. We present an analytical model for determining an optimal combination of distributed replica catalogs, catalog sizes, and replica servers. Empirical simulation results confirm the accuracy of our theoretical analysis.
Our model captures the characteristics of highly dynamic environments like peer-to-peer networks, but it can also be applied to more centralized, less dynamic grid environments like the European *DataGrid*.

1 Introduction

World wide data grids [2, 8, 4, 5] provide an unprecedented amount of aggregated storage space on geographically dispersed computers. Even when using simple commodity PC farms as compute nodes in the grid, the combined storage space may exceed hundreds of terabytes. The European particle physics laboratory *Cern*, for example, will employ approximately 50.000 PCs, grouped in clusters at various sites throughout the world, to analyze the empirical particle collision data generated by the *Large Hadron Collider* [2]. With current disk technology, the aggregated storage capacity of such a commodity data grid would easily exceed the petabyte range.

The large amount of data and its distributed and dynamic nature makes data management a big challenge. Replication, synchronization, mapping, and retrieval of data are some of the key issues.

We focus on the replication aspect in this paper. For common grid environments with unreliable components, we analyze how many replicas are needed to provide a given file availability from the view of a requester. From a system administrator's view, the model can be used to determine how many replicas are needed and how much disk space that would cost. Our model captures a peer-to-peer environment with central or distributed replica catalogs.

2 Towards Optimal Data Replication

In scientific research domains like high energy physics, data grids with thousands or millions of distributed files and thousands or millions of clients will be established over the next years. Such large grids can only be operated in a self-optimizing way with distributed data catalogs. Any central directory would inevitably become a performance bottleneck and a single point of failure.

The management of distributed data involves the following aspects:

- *replication* to improve reliability by introducing redundant file copies,

– *synchronization* for keeping replicas up-to-date,
– *mapping* to determine optimal replica locations for a fast access,
– *caching* and *prefetching* to benefit from spatial and temporal access locality,
– *staging* to improve job execution time by scheduled data transfers,
– *data movement* with efficient and secure protocols.

From these aspects, data reliability is probably the most important one for the successful uptake of grid technology in practice. Especially at this early time, the growing user community needs to gain trust in the temporal and spatial availability of their data. Clearly, file availability can be improved by creating replicas. The more replicas, the higher the availability of the file at any time. However, the required disk space grows proportionally. To find a good balance between availability and space consumption we present an analytical framework that models various architectures of replica systems. With this framework it can be calculated how many replicas are needed to provide a given availability threshold. This facilitates the design of distributed storage systems that have a similar (or better) reliability as common disk subsystems or RAID servers.

3 Analytical Model

Let's assume a replica system where several copies of the same file are stored on independent nodes which are linked by some network. The nodes holding the files and catalogs may be unreliable with a given probability.

Replica Catalog. Before being able to access a replica, a name resolution must be performed. This is done by a lookup in the *replica catalog*. Architectural choices for the catalog affect the outcome of the model: single versus multiple catalogs, unreliable versus reliable, probability of finding an entry, etc.

Access. Our analytical model distinguishes two kinds of nodes, replica nodes and catalog nodes. When accessing a file, the requester first asks one or more catalogs holding storage locators to the replicas. He then tries to access the replicas, one after the other until he succeeds. In this procedure, three things may happen:

1. there is no catalog available,
2. the catalogs do not know about any replica of the file,
3. the servers holding the replicas are down.

Global and Local View. In the analysis, it is important to distinguish between a local and global view. The *local view* is the perspective of a single component on the system when it monitors its neighboring components. In very large systems, the horizon of the local view may be limited, disclosing only a small subset of the components. The *global view* gives a birds view on all components. Typically, system designers or administrators are interested in the global view. It allows them to estimate the overall resource requirements and to configure the system according to the needs.

3.1 Basic Equations

Without loss of generality we analyze the access to a *single* file only. Accesses to multiple files can be analyzed by applying the model to each single access and combining the results.

We make use of two basic equations, the $ok(p,n)$ function and the binomial function. The $ok(p,n)$ function (eq. 1) describes the probability that at least 1 out of n redundant systems is available when the single systems have a probability p of being online.

$$\begin{aligned} &\forall p, n \in \mathbb{R}, 0 \le p \le 1, n \ge 0 : \\ &ok(p,n) := 1 - (1-p)^{\lfloor n \rfloor} \end{aligned} \quad (1)$$

The probability for a system being *not* available is $(1-p)$ and the probability for all n systems failing is $(1-p)^n, n \in \mathbb{N}$. Note that a system is either available or not. It cannot be the case that only a fraction of a system is available. To allow $n \in \mathbb{R}$ for convenience, we floor n so that we get $(1-p)^{\lfloor n \rfloor}$ as the probability that all $\lfloor n \rfloor$ systems are not available. Then $1-(1-p)^{\lfloor n \rfloor}$ is the probability that not all $\lfloor n \rfloor$ systems are not available, or, in other words, the probability that at least one system is ok.

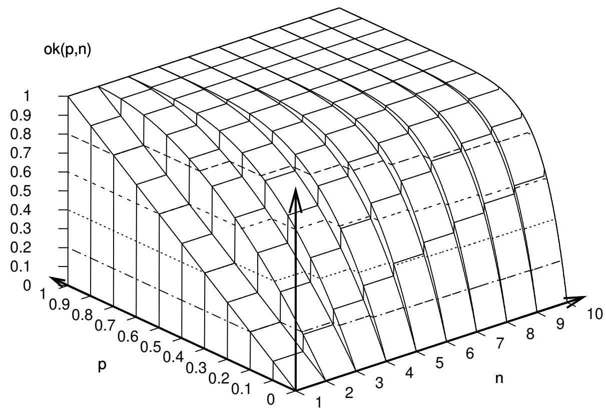


Fig. 1. $ok(p,n)$.

The $ok(p,n)$ function is illustrated in Figure 1 for different input values. The contour lines depict $z = ok(p,n)$ for $z \in \{0.2,\ 0.4,\ 0.6,\ 0.8\}$. The figure shows, for example, that $n = 8$ replicas are needed to provide a system reliability of $ok(p,n) = 0.8$ when the single components have a probability of only $p = 0.2$ to be intact.

Our second basic equation determines all combinations of k out of n items. When multiplied with the probability of their occurrence (see eq. 2), this equation can be used to determine the number of accessible replicas when there is a given number of catalogs with partly redundant (i.e. overlapping) entries. Figure 2 depicts a plot of $binom(n,k,p)$ for $n = 20$.

$$\forall n,k,p \in \mathbb{R},\ 0 \le p \le 1,\ n,k \ge 0:$$
$$binom(n,k,p) := \binom{n}{k} p^k (1-p)^{n-k} \tag{2}$$

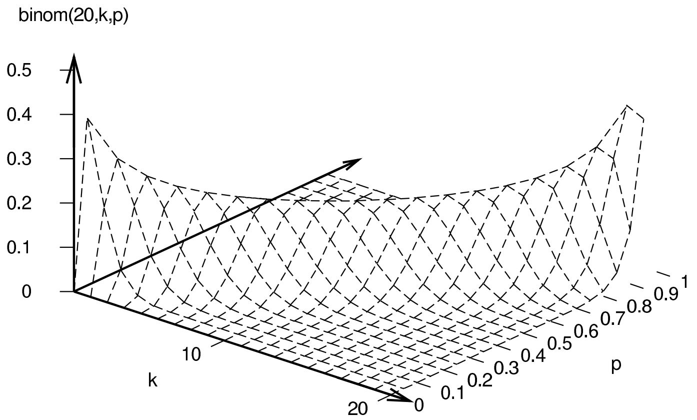


Fig. 2. $binom(n,k,p)$.

3.2 Definitions

Before presenting our analytical model, we need to define the following terms that are used throughout the analysis.

p_{rep}: *The probability for a node that holds a replica to be available.*
We do not distinguish between more or less reliable nodes. p_{rep} shall be the average uptime probability for all participating nodes holding replicas.

p_{cat}: *The probability for a node that holds a catalog to be available.*
In peer-to-peer systems, where clients also take the role of catalog servers, $p_{cat} = p_{rep}$.

p_{entry}: *The probability for a catalog to hold an entry of a given replica.*
If there are r_g replicas in the system, each catalog knows on the average about $p_{entry} \cdot r_g$ replicas. p_{entry} tells how well the catalog is informed. Replica catalogs may not be completely informed, e.g. because they have been offline or separated by the network for some time period.

r_g: *The total number of replicas of a given file in the system.*
r_g is independent of the accessibility of the replicas. We use r_g later to determine the amount of replicas needed to guarantee a certain file availability.

r_ℓ: *The number of replicas that a local node can try to access.*
Note that, in the case of multiple catalogs, r_ℓ describes only the number of non-redundant replica entries that are seen in the local view. Some listed replicas may be temporarily not accessible. For systems with just one reliable catalog $r_\ell = p_{entry} \cdot r_g$.

c**:** *The number of replica catalogs that are seen by a requester.*
Catalogs may be unavailable due to network partitioning or system downtimes. A low p_{cat} or p_{entry} may be compensated by introducing more catalogs.

3.3 Analytical Model

c	p_{cat}	**local view**	**global view**
1	100%	$ok(p_{rep}, r_\ell)$	$\sum_{i=1}^{r_g} binom(r_g, i, p_{entry}) \cdot ok(p_{rep}, i)$
c	100%	$ok(p_{rep}, r_\ell)$	$\sum_{i=1}^{r_g} binom\big(r_g, i, ok(p_{entry}, c)\big) \cdot ok(p_{rep}, i)$
1	p_{cat}	$p_{cat} \cdot ok(p_{rep}, r_\ell)$	$p_{cat} \cdot \sum_{i=1}^{r_g} binom(r_g, i, p_{entry}) \cdot ok(p_{rep}, i)$
c	p_{cat}	$ok(p_{cat}, c) \cdot ok(p_{rep}, r_\ell)$	$\sum_{j=1}^{c} binom(c, j, p_{cat}) \cdot \sum_{i=1}^{r_g} binom\big(r_g, i, ok(p_{entry}, j)\big) \cdot ok(p_{rep}, i)$

Table 1. Overview of the analytical model.

Table 1 gives an overview of the main equations of our analytical model. We distinguish between a global and a local view. The global view is that of a hypothetical totally-informed observer on the whole system, while the local view is taken from the perspective of a requester who just wants to access a file.

In the following sections, we discuss the model in more detail and show how the equations in Tab. 1 are derived.

System with one fully available catalog. First, we analyze a system with a 100% available replica catalog. This is a common situation when the catalog is installed on a server with fail-over facility and redundant network connections. Still, the catalog on this server may not be completely informed. This is the case, for example, when not all new replicas are registered or if the storage capacity is insufficient and the catalog must replace entries according to some caching strategy. The probability for a replica to be listed in the catalog is p_{entry}.

In the local view, illustrated in Fig. 3, the requester has reliable access to the catalog ($p_{cat} = 1$). He retrieves $r_\ell = 3$ replica references. But when he actually tries to access the replicas, he

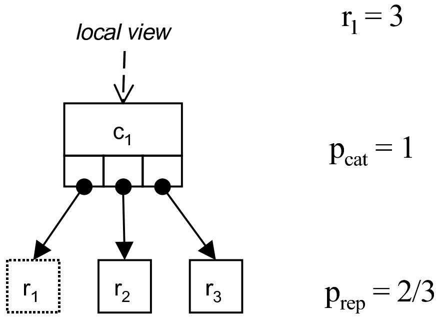


Fig. 3. One fully available catalog in the local view.

finds the first server with replica r_1 down. Only the other two replicas are currently accessible, hence $p_{rep} = 2/3$.

In general, a local requester with a fully reliable catalog can access a file with $ok(p_{rep}, r_\ell)$ probability. The success rate can be improved by increasing either the availability of the replica server p_{rep} or the number of catalog entries r_ℓ.

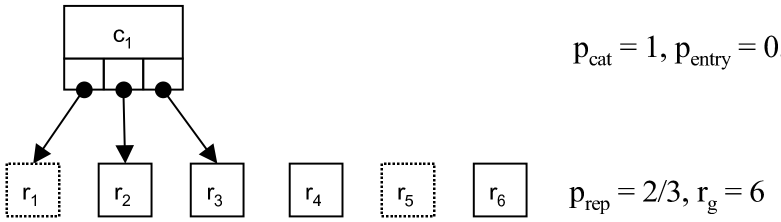


Fig. 4. One fully available catalog in the global view.

In the global view, illustrated in Fig. 4, $r_g = 6$ replicas exist. Only $p_{rep} = 2/3$ of them can be accessed, because the servers holding r_1 and r_5 are temporarily down.

In general, only those replicas that are listed in the catalog improve the overall availability, provided their server is up and running. So we first determine for all possible cases with exactly $i = 1, 2, \ldots, r_g$ replicas listed in the catalog the number of combinations and multiply the result with the corresponding likelihood of occurrence:

$$\binom{r_g}{i} {p_{entry}}^i (1 - p_{entry})^{r_g - i}$$

Summing all cases, weighted with their replica availability $ok(p_{rep}, i)$, we get the overall availability as listed in Tab. 1:

$$\sum_{i=1}^{r_g} binom(r_g, i, p_{entry}) \cdot ok(p_{rep}, i).$$

System with c fully available catalogs. We now discuss the case of c reliable catalogs instead of just one. Here, the equation for the local view does not change, because the local requester follows the same principle: He first looks into all available catalogs, determines the union of the entries and then tries to access the replicas. In the snapshot illustrated in Fig. 5 two of the six entries in the catalogs refer to the same replica, hence $r_\ell = 5$.

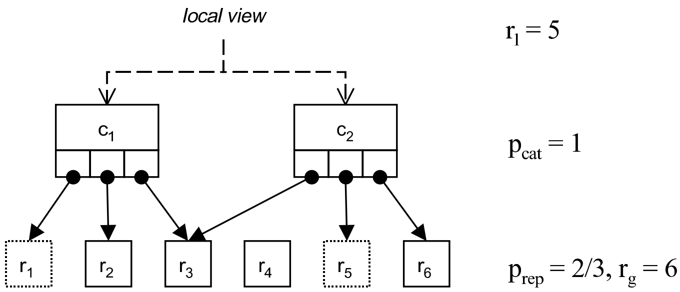


Fig. 5. Two reliable catalogs in the local view.

In the global view, we may use the same equation as above, but must allow for c catalogs instead of just one. With c catalogs, each of them holding p_{entry} entries on the average, the overall probability for a replica to be listed in at least one catalog is $ok(p_{entry}, c)$. By replacing p_{entry} in the last scenario by $ok(p_{entry}, c)$ we get

$$\sum_{i=1}^{r_g} binom\big(r_g, i, ok(p_{entry}, c)\big) \cdot ok(p_{rep}, i).$$

It is interesting to note that, for a system with two catalogs and $p_{entry} = 0.5$ our model assumes an overlap of the catalog entries that follows a binomial distribution. In other words, a 100% overlap (i.e. the two catalogs have exactly the same entries) is just as likely as a 0% overlap (i.e. both catalogs are disjunct). Most likely is a 50% overlap of the entries.

System with one catalog with availability p_{cat}. When the catalog is installed on a regular rather than a fail-safe server, we have to take a probability p_{cat} into account, that the catalog is not available. This is common for large distributed environments. In peer-to-peer systems, there is no distinction between clients and servers; each node may take the role of a client or server or even both. As a consequence, $p_{cat} = p_{rep}$.

Both, in the local and the global view, we have to multiply the above equations by p_{cat}. This is because we have to model two events in a sequence: first a lookup to an unreliable catalog, then a lookup to an unreliable replica. Hence, we get for the global view

$$p_{cat} \cdot \sum_{i=1}^{r_g} binom(r_g, i, p_{entry}) \cdot ok(p_{rep}, i).$$

As can be seen, the overall availability is restricted by the minimum of p_{cat} and the second factor. It is impossible to increase the overall availability above p_{cat} because the catalog is a single point of failure.

System with c catalogs with availability p_{cat}. With c unreliable catalogs chances of getting access to a replica are higher than when using just one unreliable catalog. In the local view a node tries to access all c catalogs, accumulates the results to r_ℓ, and then tries to access the replica in a subsequent step. Rather than p_{cat} as before, chances are now $ok(p_{cat}, c)$ to get access to an active catalog server. Hence the equation for the local view is

$$ok(p_{cat}, c) \cdot ok(p_{rep}, r_\ell).$$

Note that the availability grows with each additional visible catalog. Still, the upper limit is given by $ok(p_{cat}, c)$ for the same reasons as described above.

Depending on the number of accessible catalogs and their overlap, the listed replicas are more or less complete. To model this in the global view, we distinguish the different cases and sum all availabilities:

$$\sum_{j=1}^{c} binom(c, j, p_{cat}) \cdot \sum_{i=1}^{r_g} binom\big(r_g, i, ok(p_{entry}, j)\big) \cdot ok(p_{rep}, i).$$

4 Simulation Environment

For the empirical verification of the analytical model we implemented a simple simulator in C++. The simulator initially creates replicas in a loop and each catalog memorizes them with probability p_{entry}. In a second step, the requesters try to access a replica by asking the catalogs for known replicas. The catalogs respond with probability p_{cat}. In the positive case, the requesters try to access the replicas. Each single access is successful with probability p_{rep}. If at least one access succeeded, the file is available.

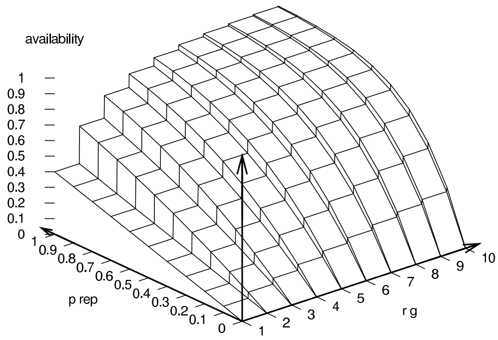
With this scheme, we determined the average file availability by running each combination of p_{rep} and r_g 100,000 times. A complete plot took only a few minutes on a PC. The results can be seen in Figs. 6 and 7.

5 Results

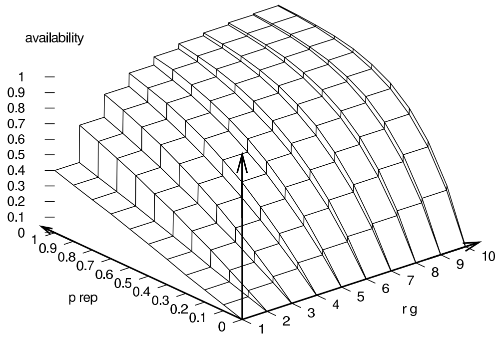
We have evaluated the accuracy of our analytical model by comparing the results to that of a simulation with the same parameter set. Figures 6 and 7 show the results. In each column (a)-(c) and (d)-(f) the analytical model is depicted at the top, the simulation in the middle, and the difference between both at the bottom.

In all four cases, the simulation matches almost ideally the results of the analytical model. The small statistical variance can be reduced by increasing the number of simulation runs.

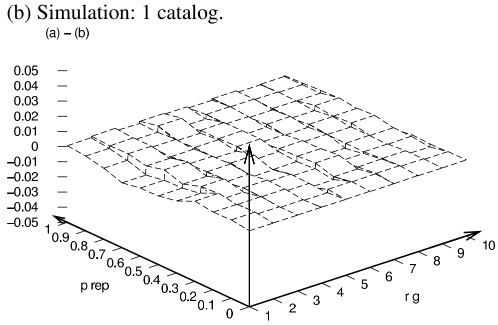
For all cases we have chosen a catalog entry probability of $p_{entry} = 0.4$. While this value may seem to be too small to properly reflect reality, we have chosen such a low probability to better illustrate the shape of the curves.



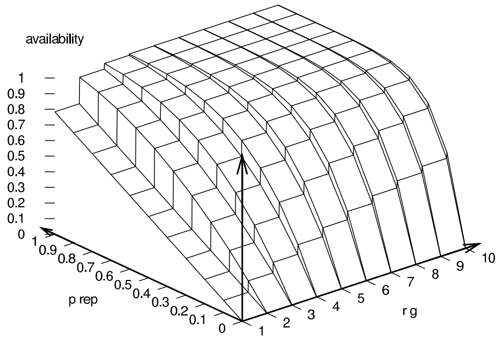
(a) Model: 1 catalog.



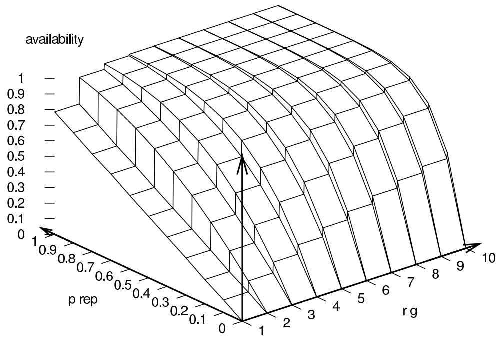
(b) Simulation: 1 catalog.



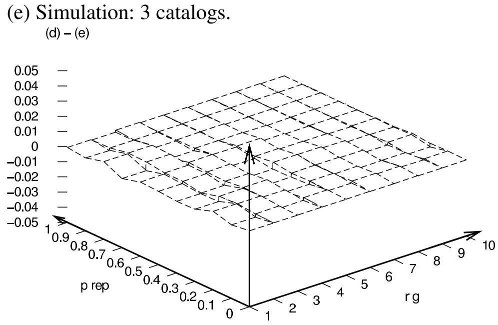
(c) Difference between (a) and (b).



(d) Model: 3 catalogs.



(e) Simulation: 3 catalogs.



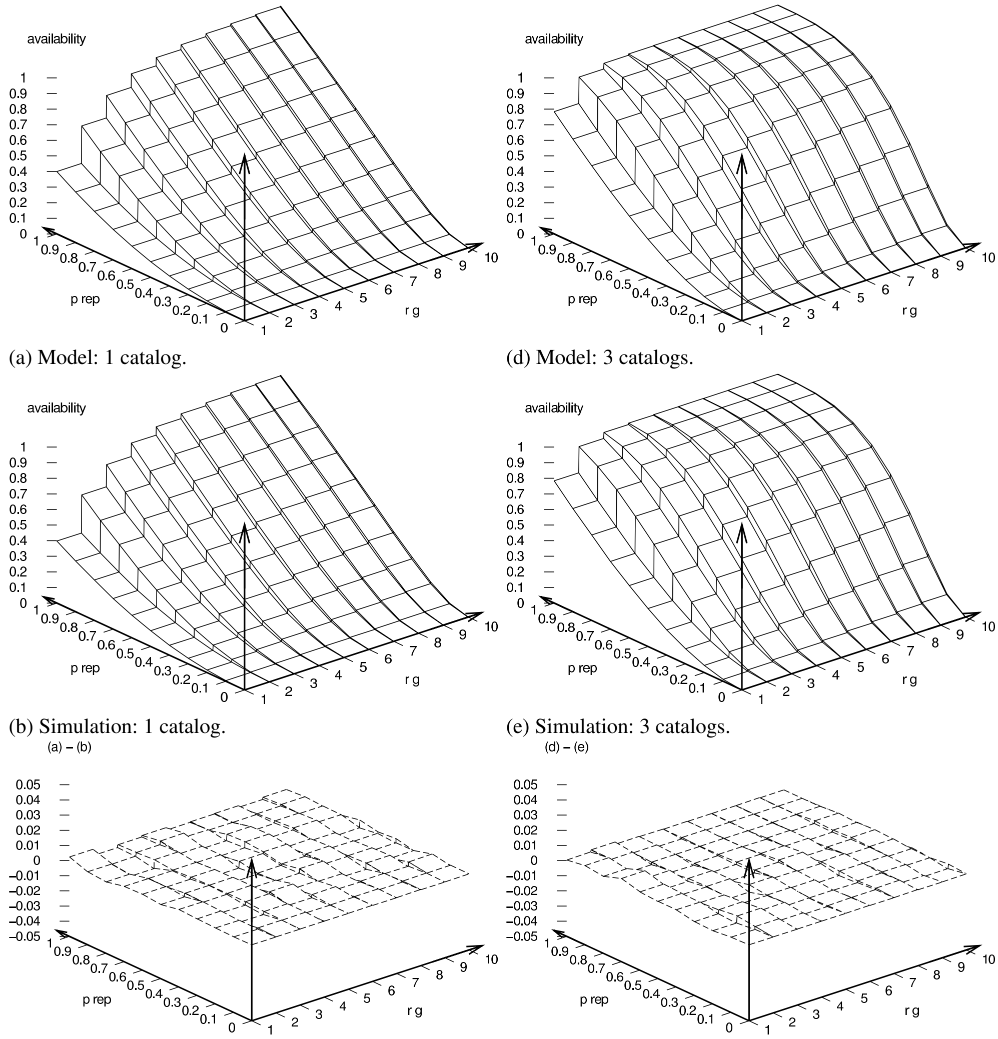
(f) Difference between (d) and (e).

Fig. 6. Analytical model versus simulation for fully available catalogs, $p_{entry} = 0.4$.

Fig. 6 shows an architecture with a catalog on a highly available server ($p_{cat} = 1$), whereas Fig. 7 depicts the results for a typical peer-to-peer system with varying $p_{cat} = p_{rep}$.

Architecture with a highly available catalog. With only one replica in the system ($r_g = 1$) a maximum file availability of 0.4 can be reached, because p_{entry} is 0.4 (Fig. 6a). When reducing the replica server availability p_{rep}, from 1.0 down to 0.0, the overall availability also decreases linearly. This linear curve in the left part of Fig. 6a changes to an asymptotic curve with an increasing number of replicas, as can be seen in the right hand side.

When holding p_{rep} fixed to some arbitrary chosen value and varying r_g, we observe how p_{entry} affects the maximum availability. The more replicas exist, the more likely it is that some of them are listed in the catalog—despite the low p_{entry}.



(a) Model: 1 catalog.
(d) Model: 3 catalogs.
(b) Simulation: 1 catalog.
(e) Simulation: 3 catalogs.
(c) Difference between (a) and (b).
(f) Difference between (d) and (e).

Fig. 7. Analytical model versus simulation for catalog availability $p_{cat} = p_{rep}$ and $p_{entry} = 0.4$.

Using three instead of one catalogs (see Fig. 6d-e), the curves get more pronounced because of the higher probability to find an entry in the merged catalogs.

Peer-to-peer architecture. Fig. 7 shows a scenario where the catalog is as unreliable as the replica servers, $p_{cat} = p_{rep}$. Therefore the maximum system availability is strictly limited by the reliability of the catalog servers. This can be seen in the curves in the right hand part of Fig. 7a, which are almost straight lines. Note that these curves are positioned slightly below an ideal straight line (not plotted), especially for small p_{rep}. This is because p_{cat}, having the same value as p_{rep}, increases the existing tendency.

With more unreliable catalogs (Fig. 7d-f) an almost arbitrary large availability can be supported. Note that this is independent of p_{rep}. In other words, a low replica availability can be compensated by more replicas and more catalogs.

	architectural parameters				r_g for maximum downtime in 70,000h of		
	p_{rep}	p_{cat}	p_{entry}	c	**1s**	**1m**	**1h**
1.	0.9	1	0.4	1	44	35	25
2.	0.9	p_{rep}	0.4	9	10	8	6
3.	0.9	p_{rep}	0.4	7	-	10	6
4.	0.9	p_{rep}	0.4	5	-	-	11
5.	0.9	1	0.9	1	9	7	6
6.	0.9	p_{rep}	0.9	9	9	7	5
7.	0.9	p_{rep}	0.9	7	-	7	5
8.	0.9	p_{rep}	0.9	5	-	-	6

Table 2. Number of required replicas r_g to meet a given file availability.

This effect is shown in Tab. 2 for more realistic parameters. Hard disk manufacturers today typically guarantee a mean time between failure of 70,000 hours (8 years). Table 2 lists the number of replicas r_g that are needed in a distributed system to achieve the same reliability, but here modeling a transient error of only one second, one minute, or one hour (see the right hand columns). These numbers are for a 90% uptime probability of the replica node and various catalog uptime probabilities of $p_{cat} = 1$ and 0.9. The number of catalogs c has been chosen to be in the one-digit range.

Under these constraints, we observe that with 9 catalogs only 10 replica are needed to guarantee that a file is only for 1 second not available during eight years of runtime (see line 2 in Tab. 2). When increasing p_{entry} from 0.4 to 0.9, only 9 replica are needed (line 6).

Note that the number of catalogs is critical here. With fewer catalogs we cannot achieve the same availability, because then the availability p_{cat} of the catalogs is the limiting factor.

6 Related Work

The impetus for this work came partly from our participation in the European *DataGrid* project for which a large-scale distributed data management environment [8] based on commodity PC farms [7] is currently built.

In addition, a recent publication of Ranganathan and Foster [6] was another stimulating factor. They describe a framework for improving availability by creating more replicas in a feedback loop. Compared to our work, they only discuss the local view of a system with one central replica catalog. Their predicted results do not match well enough to the empirical results and we found an inaccuracy in their model that makes their results somewhat unrealistic.

The adaptive data replication algorithm described in [10] focuses on caching and supporting consistency for the replicas. In contrast, we focused on the overall resource requirements to guarantee a certain file availability.

Another dynamic data replication scheme is presented in [1] that uses finite automata to predict future access patterns and then improves data locations with respect to network transfer costs. Similar approaches are used by [3] and [9].

7 Conclusion

We have presented an analytical model that describes the availability of files in replica systems with unreliable components. The correctness of the model is confirmed by our simulation results.

The model can be used to determine an optimal combination of replicas, catalogs, and catalog accuracy, while respecting constraints such as system availability and the accuracy of the catalog entries. In practice, the catalog accuracy may not only depend on the available memory space, but also on the spatial extension of the system (causing consistency overhead), temporal aspects (caching effects) and the dynamical behavior (frequency of replica movement) in the system.

Note that the goal of our work is not only to analyze existing distributed systems, but also to actively steer them. With the 'local view' scheme introduced above, single components in the system can monitor and decide about necessary activities to improve the overall file availability.

Model Extensions. Our analytical model could have been extended by considering network links as separate entities. We did not do so, because this would have unnecessarily complicated the analysis. When network breakdowns or separations occur independent from any other event, they can be easily incorporated by introducing an additional term in the equation of the component availability.

More difficult is a model extension to reflect the distribution of the namespace over several catalogs. This is commonly done in practice—especially in search engines—to avoid performance bottlenecks. In a two-catalog system, for example, one catalog holds all entries of file names starting with the letters 'a-m' and another holds files with 'n-z'. This catalog splitting must be treated separately in our model, because for accessing one file (i.e. our focus) only replica catalogs responsible for this filename will be asked. Modeling namespaces just by introducing a lower p_{entry} would give wrong results, because there would be no longer a binomial distribution of the entries in the catalogs.

Part of this work was funded by the EU DataGrid Project.

References

1. S. Acharya, S.B. Zdonik. An Efficient Scheme for Dynamic Data Replication. Brown University CS-93-43, 1993.
2. S. Bethke, et al., Report of the Steering Group of the LHC Computing Review, Technical Report CERN European Origanization for Nuclear Research, February 2001.
3. M. Carman, F. Zini, L. Serafini, K Stockinger. Towards an economy-based optimisation of file access and replication on a data grid. *In International Workshop on Agend based Cluster and Grid Computing at International Symposium on Cluster Computing and the Grid (CCGrid2002).*, pp 340–345, Berlin, Germany, May 2002. IEEE Computer Society Press.
4. E. Deelman, C. Kesselman, R.D. Williams, A. Lazzarini, T.A. Prince, J. Romano, B. Allen, A Virtual Data Grid for LIGO *Lecture Notes in Computer Science 2110*, pp. 3-12, Springer, 2002.
5. Particle Physics Data Grid (PPDG), http://www.ppdg.net/.

6. K. Ranganathan, A. Iamnitchi, I. Foster. Improving data availability through dynamic model-driven replication in large peer-to-peer communities. In *Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid2002)*, pages 376–381, IEEE Computer Society, May 2002.
7. A. Reinefeld, V. Lindenstruth. How to Build a High-Performance Compute Cluster for the Grid. In *International Workshop on Metacomputing Systems and Applications, MSA'2001*, IEEE Computer Society Press, Sept. 2001, pp. 221-227.
8. H. Stockinger, A. Samar, B. Allcock, I. Foster, K. Holtman, B. Tierney. File and Object Replication in Data Grids. *Journal of Cluster Computing*, 5(3):305-314, 2002.
9. H. Stockinger, K. Stockinger, E. Schikuta, I. Willers. Towards a Cost Model for Distributed and Replicated Data Stores. *In 9th Euromicro Workshop on Parallel and Distributed Processing PDP 2001*, Mantova, Italy, February 7-9 2001. IEEE Computer Society Press.
10. O. Wolfson, S. Jajodia, Y. Huang. An Adaptive Data Replication Algorithm. *ACM Transactions on Database Systems*, vol. 22, pp. 255–314, 1997.