Data Replication and Delay Balancing in Heterogeneous Disk Systems [*]

D. ROTEM
Lawrence Berkeley National Laboratory, USA

S. SESHADRI
Leonard N. Stern School of Business, New York University, USA

L.M. BERNARDO
Lawrence Berkeley National Laboratory, USA

Abstract

Declustering and replication are well known techniques used to improve response time of queries in parallel disk environments. As data replication incurs a penalty for updates, database designers face the problem of finding which part of the database to load on each disk and how these parts should be replicated. This problem becomes more complicated in heterogeneous environments where disks have different speeds and capacities as intuitively faster disks should be assigned a greater portion of the database to balance the delay in the system. We present analytic results and heuristics providing "near" optimal solutions to the combined problem of finding declustering proportions and replication schemes in such environments. Our model takes into account the characteristics of the disks such as speeds and capacities, as well as, access patterns to the data. Simulation results for various environments comparing different replication strategies are also provided.

Keywords: Data Replication, Heterogeneous Disk Systems

1 Introduction

Declustering data on parallel disk systems, is a well known technique for improving I/O performance [3, 5, 7, 8, 6, 9]. The idea is to distribute the data among n parallel disks so that data which is likely to be requested together by a query

[*]This work was supported by the Office of Energy Research, Office of Computational and Technology Research, Division of Mathematical, Information, and Computational Sciences, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098 and by the Defense Advanced Research Projects Agency.

is allocated to different disks. In this way, queries reading large amounts of data may experience significant improvement in their response time as each disk has to access only a fraction of the answer.

Proposed declustering methods differ from each other in the way in which they decompose the data among the disks. Methods based on hashing techniques, error-correcting codes, Hilbert curves and lattice-based decomposition have all appeared in recent research literature. All these methods attempt to balance the load among the disks as the response time is dependent on the disk with maximal delay among all the disks involved with the query.

Replication is another technique often used for improving I/O response time. Recently there have been many research efforts to improve disk access times by replicating some portion or all of the database across multiple disks [1]. Such replication methods are known as mirrored disks [11] or shadow sets [12] and are supported by many vendors of fault-tolerant systems. It is well known that data replication may lead to higher availability as alternative disks may be used in case of disk failures. In addition, I/O performance may be improved for applications that are read-intensive. The reason for this is that if a requested page is replicated on r disks, we may choose to read a copy from one of the r disks currently least utilized or the one which can service the request with minimum seek time based on the current disk arm location [1, 10]. There are also obvious disadvantages to data replication such as higher storage costs and increased update costs – each replica must be updated to maintain consistency.

Most research work on declustering and replication mentioned above was done on symmetrical disk systems. In such systems the disks have similar transfer rates and capacities, the maximum I/O speed-up is achieved if the response to a typical query is balanced across the disks, However, many applications have to use existing platforms and hardware configurations and therefore need to decluster and replicate the data over a network of heterogeneous disks and servers. The various disks in the system may differ in their transfer rates, seek times and their capacities. Furthermore, the servers speeds may differ as well.

This work is motivated by a terrain visualization application where a collection of aerial photos (called tiles) are stored on a set of m parallel disks [4]. The tiles are of different resolutions such that the same area can be viewed with more or less detail as needed. A visualization request consists of a list of tiles that must be fetched from the disks and sent to the visualization workstation. The response time for servicing each request must be minimal to allow a continuous display of the terrain without “hiccups”. We found that by declustering the tiles and then replicating some of the tiles across a subset of the disks we could greatly improve the response time for the reasons explained above. In this case we chose to replicate low resolution tiles more than others as they are requested with more frequency as well.

In [4] it was shown that declustering in heterogeneous environments could lead to greatly improved performance if the fraction of the database loaded on each disk is approximately proportional to their speed. This must be done of course subject to available capacities on the disks. The algorithm presented there produced a declusterization scheme which provides the exact fraction of the database to be loaded on each disk. In this paper we show that, similar to declustering, replication in heterogeneous environments must take into account the characteristics of the underlying disks.

As an example for the kind of problem we are looking at, consider a system with two disks F and S (fast and slow) where the fast disk is 4 times faster than the slow one. Intuitively, it seems that the loading of the disks should follow the speed ratios, *i.e.*, the faster disk should get 4 times as many records as the slow one. However, in a previous paper a HEURISTIC declustering strategy was developed where it was shown that the it is beneficial to load the faster disk with more than the proportional number of records. Details of this heuristic are given later in this paper. We consider three examples of clustering and compare them with a replication strategy:

Clustering Records are spilt between the two disks.

- EQUAL- $0.5N$ records on each of F and S.
- PROPORTIONAL- $0.8N$ on F and $0.2N$ on S.
- HEURISTIC- $0.886N$ on F and $0.114N$ on S.

Replication Records are replicated between the two disks.

- REPLICATION - N records on each of F and S.

Another strategy, which we will not discuss in our analysis, would be to consider partial replication. Note that the ratios for the HEURISTIC example, $0.886 : 0.114$, depend on the disks speed ratio $4 : 1$ and would be different for a different disks speed ratio.

Given below are the response times of the system and standard deviation under three scenarios (1000 simulations for each case). The response times are measured in terms of 1 unit of time, defined as the time to retrieve one record using the slower disk. As we can see from the above cases, EQUAL is not a good solution in heterogeneous environments. The REPLICATION strategy typically has a faster response time and smaller standard deviation than the other strategies. However the advantage of replication decreases with higher update rates and query sizes. Although the above tradeoffs between declustering and replication are not surprising, it is very difficult to predict the values of the above parameters at which one strategy will outperform the others. In the remaining sections of this paper we provide a detailed comparative analysis and simulation in order to determine under what situations is replication better than declustering or vice versa.

Our main results are that,

Average	Std. Devn.	Strategy
0.809992	0.316107	REPLICATION
1.21200	0.474401	PROPORTIONAL
2.05600	0.918076	EQUAL
1.05700	0.244440	HEURISTIC

Table 1: Case 1. Query size 4 records - No updates.

Average	Std. Devn.	Strategy
1.68802	1.66236	REPLICATION
1.44575	0.670909	PROPORTIONAL
2.48975	1.30718	EQUAL
1.25850	0.465889	HEURISTIC

Table 2: Case 2. Query size 4. Update probability 0.2, each affecting 4 records.

Average	Std. Devn.	Strategy
20.0008	2.40797E-02	REPLICATION
21.9920	2.18781	PROPORTIONAL
49.7850	5.10498	EQUAL
21.3092	1.19710	HEURISTIC

Table 3: Case 3. Query size 1000 records. No updates.

1. We provide a method for modeling disk retrieval times under declustering and replication in the presence of updates.

2. This method produces approximate formulae and bounds for the expected retrieval time, and the error of approximation is shown to be small using simulation.

3. Using our formulae, it can be inferred that replication is favored to declustering when the update rates are small, the number of disks large and the size of the query small.

4. The value of improved retrieval time due to replication is inversely proportional to the update rate, and proportional to the square root of the ratio of number of disks divided by the average size of the query.

5. In heterogeneous environments, declustering may yield almost as good performance as replication. Moreover, the expected retrieval time is often governed by the performance of just the fastest disks.

The organization of the paper is as follows: In Section 2 we provide some preliminary analysis for the case of two disks. In Section 3 this is extended to the case of multiple disks. In Section 4 some simulation results and error analysis is presented to validate our analytic formulae. Finally, in Section 5 some conclusions and directions for future work are provided. To make this paper self contained, a derivation of the optimal declustering formula from [4] is given in Appendix A.

2 Two Disks

In this section, we develop a modeling methodology for comparing replication versus declustering when there are just two disks. The formulae for the performance measures are nearly exact given our modeling assumptions listed below. The expression for the retrieval time when there are no updates are derived first. In the final part of this section, we develop a robust modeling methodology for incorporating updates. The notation employed is summarized below.

B_i	Speed of Disk i
R_D	Retrieval time of a query under declustering. $R_D(B_1, .., B_m, p_1, ...p_{m-1}, N, \gamma)$ is the retrieval time when the size of the request is N, the declustering is done in the proportions p_i on disk i with speed B_i, and updates bring a work load of γ per unit time. m is the number of disks.
R_R	Retrieval time of query under replication. $R_R(B_1, ..., B_m, N, \gamma)$ is the retrieval time when the database is fully replicated on the m disks, with speeds B_i, and updates bring a work load of γ per unit time. The size of the request is N.
N	Size of query
γ	Workload brought per unit time by updates on the full database
$\phi(x)$	Standard Normal Density function evaluated at x
$\Phi(x)$	Standard Normal Distribution function evaluated at x
$\Phi^c(x)$	Complement of $\Phi(x)$, (i.e., $1 - \Phi(x)$)
Unit Load	One that will take $1/B$ units of time to perform using a disk of speed B.
E[.]	Expected value

Table 4: Notation Used in the Paper

2.1 Performance without Updates

We first consider optimal declustering. We are given two disks, with speeds B_1 and B_2, with $B_1 > B_2$. A total of N records are to be retrieved by a query. Assume that a fraction p of the records are placed on the faster disk with speed B_1, and the remaining records on the other disk. The number of records requested from the faster disk is a random variable given by the Binomial distribution with parameters (N, p). We will approximate this distribution using a random variable, X, distributed normally with mean Np and standard deviation $\sqrt{Np(1-p)}$. Then the number of records requested from the two disks can be approximated using X, and $N - X$.

Let $\alpha = NB_1/(B_1 + B_2)$, $\mu = Np$, and $\sigma = \sqrt{Np(1-p)}$. The decision variable is p, the fraction of records to place on the faster disk. The expected retrieval time for the N records can be written as (see Appendix):

$$\begin{aligned} E[R_D(B_1, B_2, p, N, 0)] \approx{}& \frac{N}{B_1 + B_2} \\ &+(\mu - \alpha)\left(-\frac{1}{B_2}\Phi\left(\frac{\alpha-\mu}{\sigma}\right) + \frac{1}{B_1}\Phi^c\left(\frac{\alpha-\mu}{\sigma}\right)\right) \\ &+\sigma\phi\left(\frac{\alpha-\mu}{\sigma}\right)\left(\frac{1}{B_1} + \frac{1}{B_2}\right). \end{aligned} \tag{1}$$

Proposition 1 *An approximate value for p^*, the optimal value of p to obtain the minimum expected retrieval time, is given by solving*

$$\Phi^c\left(\frac{\alpha - \mu^*}{\sigma^*}\right) = \frac{B_1}{B_1 + B_2}. \tag{2}$$

Proof: See Appendix.

In equation (2), the optimal value of μ and σ are denoted using an asterisk, as they depend on the optimal declustering proportion, p^* as given below:

$$\begin{aligned} \mu^* &= Np^*, \\ \sigma^* &= \sqrt{Np^*(1-p^*)}. \end{aligned}$$

We are now in a position to compare declustering versus replication using these results.

Proposition 2 *We are given two disks with speeds, B_1 and B_2. There are no updates, the size of a query is N, and that the fraction obtained by solving equation (2) is placed on the faster disk with speed B_1. Then:*

1. *The optimal expected retrieval time is given by,*

$$E[R_D^*(B_1, B_2, p^*, N, 0)] \approx \frac{N}{B_1 + B_2} + \sigma^*\phi\left(\frac{\alpha - \mu^*}{\sigma^*}\right)\left(\frac{1}{B_1} + \frac{1}{B_2}\right).$$

2. *The expected retrieval time when the database is fully replicated on the two disks is given by,*

$$E[R_R(B_1, B_2, N, 0)] = \frac{N}{B_1 + B_2}.$$

Proof: Using equations (1) and (2), the minimum expected retrieval time when the database is optimally declustered can be expressed as,

$$\begin{aligned} E[R^*_D(B_1, B_2, p^*, N, 0)] &\approx \frac{N}{B_1 + B_2} \\ &+(\mu^* - \alpha)\left(-\frac{B_2}{(B_1 + B_2)B_2} + \frac{B_1}{(B_1 + B_2)B_1)}\right) \\ &+\sigma^*\phi\left(\frac{\alpha - \mu^*}{\sigma^*}\right)\left(\frac{1}{B_1} + \frac{1}{B_2}\right) \\ &= \frac{N}{B_1 + B_2} + \sigma^*\phi\left(\frac{\alpha - \mu^*}{\sigma^*}\right)\left(\frac{1}{B_1} + \frac{1}{B_2}\right). \end{aligned} \tag{3}$$

When the database is fully replicated, the two disks work in tandem to retrieve the data. Therefore the effective retrieval speed is $B_1 + B_2$. It follows that the first term on the right hand side of equation (3), $N/(B_1 + B_2)$ corresponds to the retrieval time in a fully replicated database. □

In equation (3), the last term on the right hand side

$$\sigma^*\phi\left(\frac{\alpha - \mu^*}{\sigma^*}\right)\left(\frac{1}{B_1} + \frac{1}{B_2}\right),$$

is referred to in the sequel as the "extra" retrieval time arising due to randomness. The randomness itself is the result of splitting a request across a declustered database. It follows that if there were no updates, replication would outperform declustering due to this extra term in equation (3). For the sequel, it is also useful to note that the value of $\phi((\alpha - \mu^*)/\sigma^*)$ depends only on the ratio $B_1/(B_1 + B_2)$ and not on the size of the query N. In the next section we incorporate updates.

2.2 Modeling Updates

We now extend the formulae given in the previous section to incorporate updates. We make a key modeling assumption that updates result in the slowing down of the disks. This modeling assumption will be denoted as SD. The assumption SD is valid for example if the updates have preemptive priority over retrieval, or updates take a relatively short time compared to retrievals. We examine the error from employing the assumption SD using simulations, in section 4. We shall develop equations for measuring the magnitude of slow down in disk speeds due to updates and apply this slow down factor to determine the retrieval time. For a similar modeling approach, the reader is referred to section 2.4.2 on modeling Ancillary work given in [2].

We are given two disks with speeds B_1 and B_2, and that the optimal declustering proportion p^* of records has been placed on the disk with speed B_1. We are given that updates create a load of γ units of work per unit time. When the data is declustered, assume that the update work created for the two disks is in the proportion, $p^* : (1 - p^*)$, and that the disk speeds are reduced due to this extra work.

Proposition 3 *(1) The effective (slowed down) speed of the two disks are given by*

$$B_1(1 - \frac{\gamma}{B_1 + B_2}) \quad \textit{and} \quad B_2(1 - \frac{\gamma}{B_1 + B_2}).$$

(2) The expected retrieval time when optimally declustered is given by,

$$E[R_D^*(B_1, B_2, p^*, N, \gamma)] \approx$$
$$\approx \left(1 - \frac{\gamma}{B_1 + B_2}\right)^{-1} \left(\frac{N}{B_1 + B_2} + \sigma^* \phi\left(\frac{\alpha - \mu^*}{\sigma^*}\right)\left(\frac{1}{B_1} + \frac{1}{B_2}\right)\right).$$

Proof: As defined, a unit load of work for a disk of speed B is one that will take $1/B$ units of time to perform. We are also given that updates bring a “load” of γ units of work per unit time. Due to the fact that the records are declustered using the optimal declustering fraction p^*, updates result in a load of γp^* of work per unit time for the disk with speed B_1. The rest of the updating work goes to the other disk. Consider the interval of time $[0, T]$. In this interval, the amount of time devoted to updates by the first disk with speed of B_1 is given by, $p^* \gamma T / B_1$. The time available to retrieve records is given by, $T(1 - p^* \gamma / B_1)$. Thus only $B_1 \times T(1 - p^* \gamma / B_1) = T(B_1 - p^* \gamma)$ records can be retrieved in time T. In practice, see equation (2), $p^* \approx B_1/(B_1 + B_2)$. Thus the effective disk speeds are given by,

$$(B_1 - p^* \gamma) \approx B_1(1 - \frac{\gamma}{B_1 + B_2}) \tag{4}$$

$$(B_2 - (1 - p^*)\gamma) \approx B_2(1 - \frac{\gamma}{B_1 + B_2}) \tag{5}$$

Equations (4) and (5), indicate that *both* disks are slowed down by the *same* factor of $(1 - \gamma/(B_1 + B_2))$. Dividing the expected retrieval time given in equation (3) by this factor yields the approximation,

$$E[R_D^*(B_1, B_2, p^*, N, \gamma)] \approx \tag{6}$$
$$\approx \left(1 - \frac{\gamma}{B_1 + B_2}\right)^{-1} \left(\frac{N}{B_1 + B_2} + \sigma^* \phi\left(\frac{\alpha - \mu^*}{\sigma^*}\right)\left(\frac{1}{B_1} + \frac{1}{B_2}\right)\right).$$

□

Proposition 4 *We are given two disks with speeds B_1 and B_2, and that the records are fully replicated on the two disks. We are also given that updates create a load of γ units of work per unit time. Then the expected retrieval time is given by,*

$$E[R_R(B_1,...,B_m,N,\gamma)] \approx \frac{N}{(B_1-\gamma)+(B_2-\gamma)}.$$

Proof: Any update will affect both disks. The amount of work that can be done in an interval of time $[0,T]$ by a disk of speed B, will be given by $T(B-\gamma)$ (see Proposition 3 for the details). Thus the effective (slowed down) speed of the disk will be given by $(B-\gamma)$. Finally in order to minimize the retrieval time when the data is completely replicated, assume that we retrieve records in $proportion$ to these slowed down rates. It then follows that we obtain a combined speed of $(B_1-\gamma)+(B_2-\gamma)$ for addressing the query. Therefore the query can be addressed in the time,

$$E[R_R(B_1,...,B_m,N,\gamma)] \approx \frac{N}{(B_1-\gamma)+(B_2-\gamma)}. \tag{7}$$

□

Equations (6) and (7) can now be combined to yield the following proposition.

Proposition 5 *Under the assumptions stated in Propositions 3 and 4, the ratio of the expected retrieval time under declustering to that under replication is given by,*

$$\frac{E[R_D^*(B_1,B_2,p^*,N,\gamma)]}{E[R_R(B_1,...,B_m,N,\gamma)]} \approx (1+\sqrt{\frac{p^*(1-p^*)}{N}})\frac{(B_1+B_2)^2}{(B_1B_2)}\frac{(B_1+B_2-2\gamma)}{(B_1+B_2-\gamma)}. \tag{8}$$

Proof: Straightforward. □

The proposition indicates that declustering is favored when the size of the request N or the rate of updates, γ, is large. Some numerical results illustrating the two phenomena are given in Figure 1. In this example, there are two disks, and their speeds are (8,1) in one case, and (4.5,4.5) in the other. As a benchmark to compare the effect of optimal declustering we have also provided the expected retrieval time when the records are declustered in proportion to the speeds of the two disk, this is termed *proportional allocation*. The expected retrieval time under proportional allocation (declustering) has been labeled as P, the time under the optimal declustering as H, and the time under replication as R. We see from the figure that,

1. The improvement obtained by using the optimal proportions (i.e., using the value of p given by (2)) for declustering (denoted as H) over the proportional allocation (labeled P) can be significant when the request size is small, and the disk speeds are quite different.

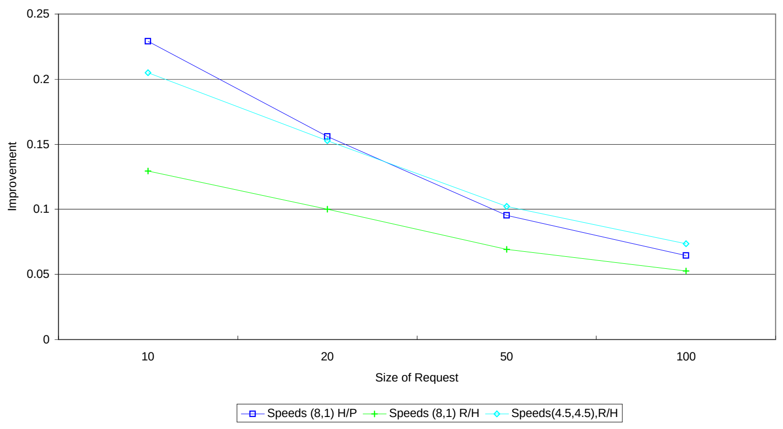

Figure 1: Effect of the Size of the Query (No Updates).

2. The improvement due to replication (R) is greater when the disk speeds are equal. This phenomenon occurs because both under H and R, the faster disk takes most of the load when the two disks are unbalanced.

3. We note too that the improvement falls quite sharply (at the rate of $1/\sqrt{N}$ from equation (8)) with the size of the request.

3 Multiple Disks

In the case of multiple disks, there is no closed form expression similar to equation (2) that is available for determining the declustering proportions. A heuristic procedure based on equation (2) can be used to decluster the data when there are multiple disks, as shown in [4]. The procedure is given below.

Procedure HEURISTIC:

```
Step 0: Read number of disks ($m$), disk speeds ($B_i$) and
average number records retrieved ($n$). Read tolerance ($tol$).
We assume that disk speeds are ordered as: $B_1 > B_2 .. > B_m$.
Set proportions to be allocated on disk $i$ as
$p_i = B_i/(B_1 + B_2 + \cdots + B_m)$. Set $error = \epsilon$.
Step 1:    do while ($error > tol$)
               $error = 0.0$
               do $i = 1, m - 1$
               $N = (p_i + p_{i+1}) * n$
               $b_1 = B_i$
               $b_2 = B_{i+1}$
```

```
        Solve for $p$ in (2) using a search procedure
        $temp = p_i$
        $p_i = p * N/nrec$
        $p_{i+1} = (1-p) * N/nrec$
        $error = error + |temp - p_i|$
        enddo
    enddo
```

In this procedure, the allocation is initialized to be proportional to the disk speeds in Step 0. Then the records are reallocated using equation (2) beginning with the two fastest disks, then reallocated on the next two fastest, and so forth. Usually the procedure converges in 10-20 iterations of Step 1.1. It can be proved that this procedure will eventually stop. Using this procedure, the fastest disk is favored with an allocation of records in a proportion greater than its speed, see Step 1.1. This situation is exactly as it was in the case of two disks. In that case, equation (2) shows that the proportion of work for the faster disk, p^* will be greater than simply the ratio of speeds $B_1/(B_1+B_2)$.

Because of this bias in the allocation to the faster disks, we have found that in an heterogeneous environment the retrieval time is basically governed by the allocation to the two $fastest$ disks with speeds B_1, and B_2. The comparison of the retrieval times under declustering with respect to complete replication on all disks can be carried out using this allocation procedure.

Consider m disks, with speeds, $B_1 \geq B_2 \geq \ldots \geq B_m$. Assume that the records are allocated using the procedure HEURISTIC. Each query consists of N records to be retrieved, and updates bring a load of γ of work per unit time. Assume that the errors introduced due to the Normal approximation to the Binomial, as well as due to SD (modeling the effect of updates as "slowing down" of the disks) are negligible. Let α_1, μ_1^*, σ_1^* correspond to the allocation produced by the HEURISTIC for the two fastest disks with speeds B_1, B_2 as follows. Let p_1^*, p_2^* be the number of records allocated to the fastest two disks under the heuristic procedure. Thus a total of $N_1 = N(p_1^* + p_2^*)$ records are allocated to these two disks. Let $\alpha_1 = B_1/(B_1+B_2)$, and $p = p_1^*/(p_1^*+p_2^*)$. Denote, $\sigma_1^* = \sqrt{N_1 p(1-p)}$, $\mu_1^* = N_1 p$. Similarly, the values of α_j, μ_j, and σ_j, correspond to the disk pair $(j, j+1)$.

Proposition 6 *(1) A lower bound on the expected retrieval time is given by,*

$$E[R_D^*(B_1,\ldots,B_m,p_1^*,\ldots,p_{m-1}^*,N,\gamma)] \leq \left(1 - \frac{\gamma}{B_1+\cdots+B_m}\right)^{-1} \times \left(\frac{N}{B_1+\cdots+B_m} + \sigma_1^*\phi\left(\frac{\alpha_1-\mu_1^*}{\sigma_1^*}\right)\left(\frac{1}{B_1}+\frac{1}{B_2}\right)\right). \quad (9)$$

(2) If m is an even number, then an upper bound for the expected retrieval time is

given by,

$$\left(1 - \frac{\gamma}{B_1 + \cdots + B_m}\right)^{-1} \times$$
$$\times \left(\frac{N}{B_1 + \cdots + B_m} + \sum_{j \ odd} \sigma_j^* \phi \left(\frac{\alpha_j - \mu_j^*}{\sigma_j^*} \right) \left(\frac{1}{B_j} + \frac{1}{B_{j+1}} \right) \right).$$

(3) The retrieval time under complete replication is given by,

$$E[R_R(B_1, B_2, ..., B_m, N, \gamma)] \approx \frac{N}{(B_1 - \gamma) + \cdots + (B_m - \gamma)}. \qquad (10)$$

Proof: (1) The term,

$$\left(1 - \frac{\gamma}{B_1 + \cdots + B_m}\right)^{-1},$$

in equation (9), incorporates the effect of updates (compare with equation (6)). The term $N/(B_1 + ... + B_m)$ in (9) is simply the deterministic component of the retrieval time. This retrieval time is increased due to randomness induced by the declustering. The increase due to randomness has been estimated in equation (9) using the term

$$\sigma_1^* \phi \left(\frac{\alpha_1 - \mu_1^*}{\sigma_1^*} \right) \left(\frac{1}{B_1} + \frac{1}{B_2} \right).$$

This correction term however only accounts for the randomness in retrieval times between the first two disks (see Proposition 2), whereas there is the randomness due to splitting the request over the other disks which has not been accounted for in this formula. Therefore we get a lower bound for the retrieval time, if the errors introduced by the other assumptions are not significant. Algebraically, if X_i is the random retrieval time from disk i, then, this fact can be expressed as $\max(X_1, X_2, ..., X_m) \geq \max(X_1, X_2)$.

(2) The upper bound has been constructed using a similar logic. In the expression for the upper bound, we take each successive pair of disks, (1,2), (3,4) .. ($m-1$,m), and simply add the “extra” retrieval time due to randomness, arising from each pair of disks. We get an upper bound, because the actual extra time due to randomness is the maximum of these pairs of values. Algebraically, if X_i is the random retrieval time from disk i, then, this fact can be expressed as $\max(X_1, X_2, .., X_m) \leq \max(X_1, X_2) + \max(X_3, X_4) + \cdots + \max(X_{m-1}, X_m)$.

(3) The retrieval time under the replication strategy is similar to the one given by equation (7), and the logic for modeling the effect of updates is similar. $\square$

We provide two graphs, Figures 2 and 3, that depict the percentage error in using equation (9) for the expected retrieval time. As can be seen from these

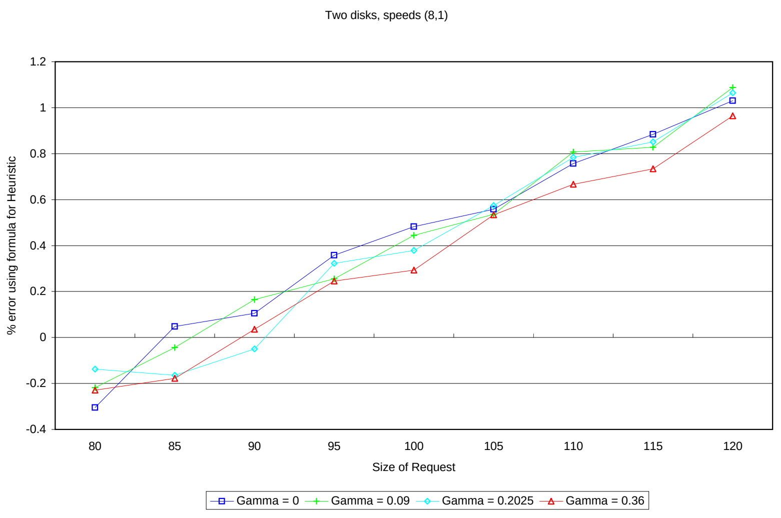

Figure 2: Error Analysis in Modeling Updates, Speeds 8, 1

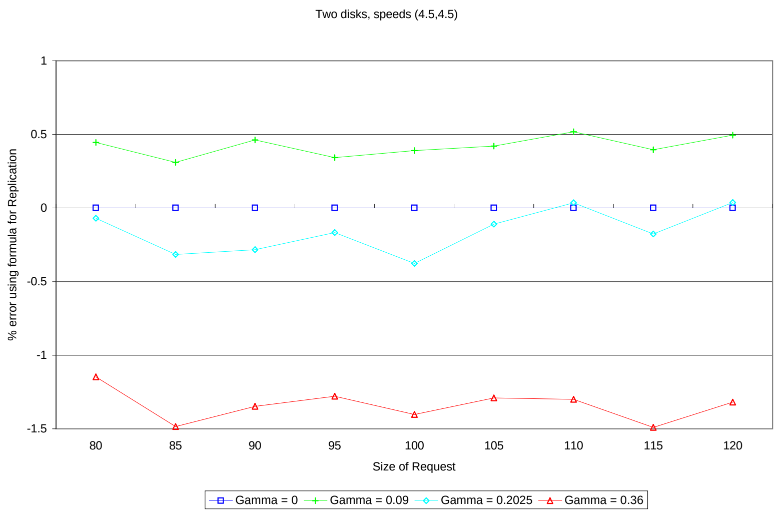

Figure 3: Error Analysis in Modeling Updates, Speeds 4.5, 4.5

figures, the combined error of approximation as well as due to our modeling assumptions is less than 2 $\%$ for this two disk case. In contrast we have found that the upper bound is sharper (with error less than 1 $\%$ in all cases tested) compared to the the lower bound. We use only the lower bound below, to emphasize the fact that with heterogeneous disks, the "extra" retrieval time due to randomness is governed by the two fastest disks. The error due to using the lower bound for the actual retrieval time will be discussed below. From equation (10), replication becomes less attractive with increasing number of disks, because the effect of updates becomes much more severe. We also see that if $B_m \approx \gamma$, then the mth disk does not contribute to speedier retrieval. We summarize these results into a single proposition,

Proposition 7 *Given that there are m disks with speeds $B_i, i = 1, 2, 3, ..., m$, and updates bring work of γ per unit of time, then declustering will be preferred to complete replication when,*

1. *The update rate γ increases;*
2. *The ratio $\frac{1}{\sqrt{N}} \times \frac{\sqrt{\sum_i B_i \times (B_1+B_2)}}{\sqrt{B_1 B_2}}$ decreases.*

Proof: As noted after Proposition 3, the value of $\phi\left((\alpha - \mu^*)/\sigma^*\right)$, does not depend on N, and can be denoted as ϕ. For practical purposes (see equation (2)), p_i^*, $i = 1, 2$ is equal to $B_i/(B_1 + \cdots + B_m)$. Therefore, the ratio of the random component to the deterministic component of the retrieval time in equation (9) is given by

$$\begin{aligned} \frac{\sigma^*\left(\frac{1}{B_1} + \frac{1}{B_2}\right)}{\frac{N}{\sum_i B_i}} &\approx \frac{\sqrt{\left(N\frac{B_1+B_2}{\sum_i B_i}\right)/B_1 B_2}}{\frac{N}{\sum_i B_i}}\phi \\ &= \frac{1}{\sqrt{N}} \times \frac{\sqrt{\sum_i B_i \times (B_1+B_2)}}{\sqrt{B_1 B_2}}\phi. \qquad (11) \end{aligned}$$

Declustering gets more attractive when the random component arising due to the random splitting of a request across disks becomes small relative to the deterministic component $N/(B_1 + \cdots + B_m)$. Therefore the smaller is the ratio in equation (11), the greater will be the preference for declustering. $\square$

The ratio in the second part of this proposition will tend to grow with the number of disks. For example, when the disks are balanced, the ratio is proportional to $\sqrt{m/N}$. If update rates are not significant, then simply the variability in retrieval across several disks could favor replication. This effect becomes significant when N is small.

4 Simulations

We have carried out simulations to test the validity of these conclusions as well as the modeling assumption SD. In these simulations, an update arrives with a fixed probability q_u along with a retrieval. The updates have to be carried out before addressing the query. Each update is assumed to affect a fixed number of records, u. When the data is declustered, the locations of the records to be updated are determined using the declustering proportions. When the data is replicated, we compute the time needed to carry out the update on each disk. We then decide which set of disks will be used to address the query using the following formula. The disk speeds are $B_1 \geq B_2 \geq \cdots \geq B_m$. The size of the query is N. Let,

$$J \quad = \quad \arg\min_j \left\{ \frac{N}{\sum\limits_{i=1}^{j} B_j} + \frac{u}{B_j} \right\}. \tag{12}$$

The fastest J disks are then used to retrieve the N records. The reader should note that this decision rule is the most effective in retrieving the requested records. It can be shown that the performance using this rule will under general conditions correspond to the the expected retrieval time obtained using the slowing down assumption (SD). The value of γ, *i.e.*, the work load due to updates per unit time can be approximated by $(B_1 + \cdots + B_m) \times uq_u$. The reason is that the disks can do work at the rate of $B_1 + \cdots + B_m$, and update work of u arises with probability q_u.

4.1 Effect of the Number of Disks

The results comparing replication versus declustering are shown in Figure 4. There, we depict the effect of the number of disks. In this example, the total speed is 120 split equally between 2, 3, 4 and 5 disks. The error in using the approximations for this example is shown in Figure 5. The percentage error of approximation increases with the number of disks. This is only to be expected, because the lower bound is based on just the two fastest disks, see equation (9), whereas in this example the disk speeds are balanced. The upper bound for the expected retrieval time is sharper than the lower bound for all cases examined in this section. The lower bound as an approximation has greater value in a heterogeneous environment, see below for an example.

In the absence of updates, replication gives greater benefits when there are more disks. The reason, see Proposition 7, is that there is increasing variability in the retrieval times when the request is split randomly between multiple disks. The effect of this phenomenon is roughly in the proportion $\sqrt{m}$, where m is the number of disks, as can be inferred from equation (11).

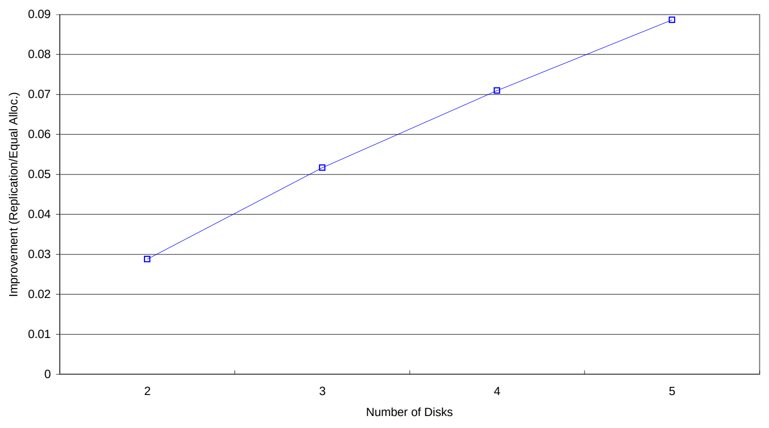

Figure 4: Replication is Favored with Increasing Number of Disks.

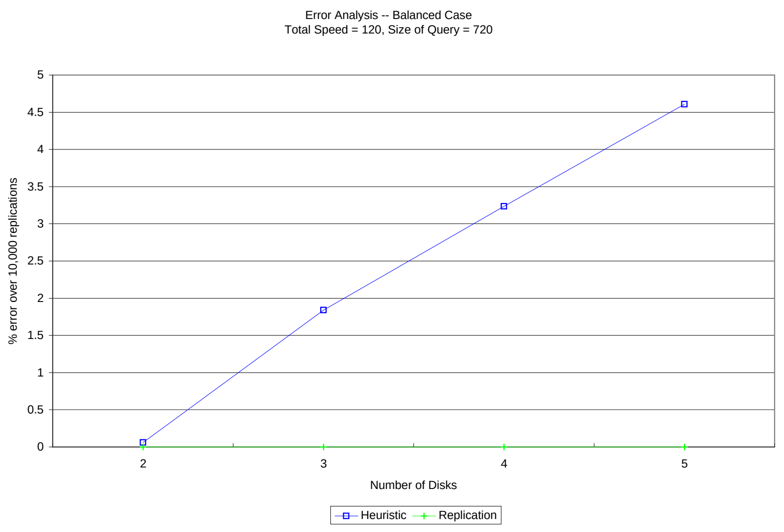

Figure 5: Error Analysis: Balanced Case, Total Speed = 120

4.2 Effect of Updates

Next we look at the effect of updates. As determined by equations (9) and (10), we see from Figure 6, that updates can diminish the value of replication. As a rule,

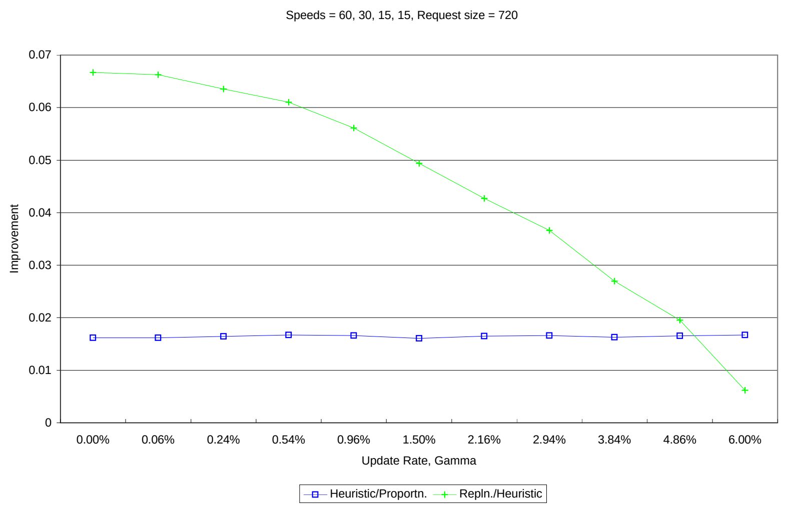

Figure 6: Declustering becomes more Attractive with Updates

the break-even for using replication occurs when the slowest disk is infrequently used ($B_m \approx \gamma$), see equation (10). In Figures 7 and 8, we show the error in using the formulae given in equations (9) and (10). The percentage error is small even for a four disk example, and actually decreases with increasing rate of updates – confirming that the use of the lower bound *as well as* the use of SD, do not detract from the quality of the approximation.

5 Conclusions and Future Work

The current work provides an insight into the tradeoffs between replication and declustering in heterogeneous environments. We showed that declustering introduces an additive random factor to the response time as compared with replication. On the other hand replication introduces a more severe penalty under updates as compared with declustering.

Our analytical results and simulations indicate the following:

- The effectiveness of declustering vs. replication improves with large request sizes. The improvement is proportional to the square root of the request size.

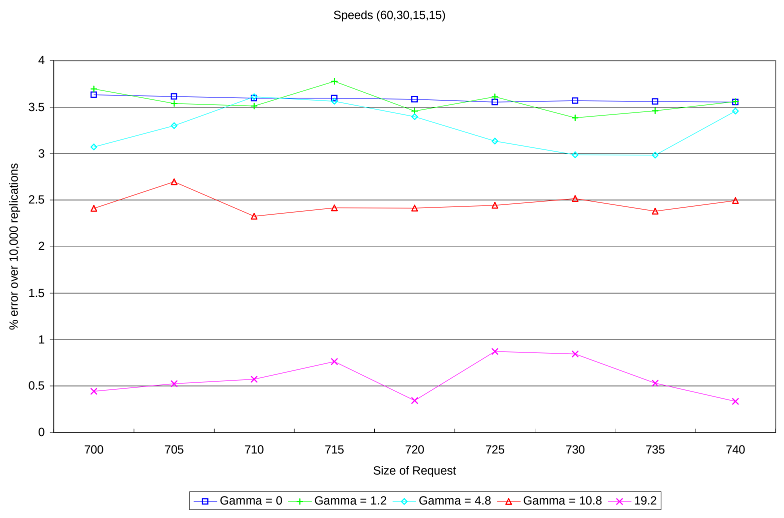

Figure 7: Error Analysis: Declustering and Updates

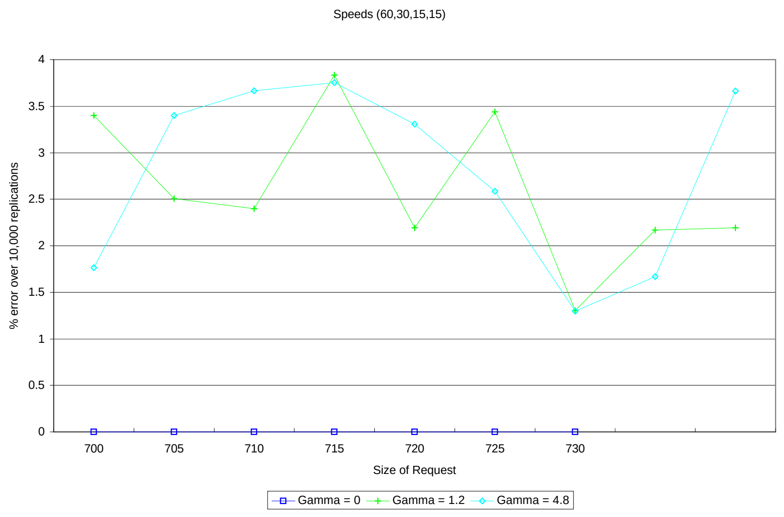

Figure 8: Error Analysis: Replication and Updates

- The effectiveness of replication is decreased linearly with the relative proportion of updates.
- The effectiveness of replication is increased with the number of available disks. This improvement is proportional to the square root of the number of disks.
- In environments with a large degree of heterogeneity among the disks, declustering is more effective as the fastest disks tend to dominate the response time and the ”randomness” due to declustering is decreased.

In the current paper we did not account for queueing effects which may affect both declustering and replication. Although we believe that the above trends will still hold, we plan to perform some future work on such analysis. Another area of interest for future work is the design and analysis of combined declustering and replication schemes. In realistic environments, a better solution may be obtained by declustering the database into chunks and then replicating some (or all) of these chunks with different degrees of multiplicity based on their access requirements.

Dorom Rotem is a member of the Scientific Data Management Research Group at Lawrence Berkeley National Laboratory, Berkeley, CA, USA. E-mail: D_Rotem@lbl.gov

Sridhar Seshadri is a faculty member at the Leonard N. Stern School of Business, New York University, New York, NY, USA. E-mail: sseshadr@stern.nyu.edu

Luis M. Bernardo is a member of the Scientific Data Management Research Group at Lawrence Berkeley National Laboratory, Berkeley, CA, USA. E-mail: LMBernardo@lbl.gov

A Appendix

Derivation of Equation (1): Consider a random variable, X, distributed normally with mean Np and standard deviation $\sqrt{Np(1-p)}$. Then,

$$\begin{aligned}
E[R_D(B_1, B_2, p, N, 0)] &= E\left(\max\left\{\frac{X}{B_1}, \frac{N-X}{B_2}\right\}\right) \\
&= E\left(\frac{X}{B_1} I\left\{X \geq \frac{NB_1}{B_1+B_2}\right\}\right) + E\left(\frac{N-X}{B_2} I\left\{X \leq \frac{NB_1}{B_1+B_2}\right\}\right) \\
&= \frac{1}{B_1} E\left(\left(X - \frac{NB_1}{B_1+B_2}\right) I\left\{X \geq \frac{NB_1}{B_1+B_2}\right\}\right) \\
&\quad + \frac{N}{B_1+B_2} P\left(X \geq \frac{NB_1}{B_1+B_2}\right)
\end{aligned}$$

$$
\begin{aligned}
&+\frac{1}{B_2}E\left(\left(\frac{NB_1}{B_1+B_2}-X\right)I\left\{X\le\frac{NB_1}{B_1+B_2}\right\}\right)\\
&+\frac{1}{B_2}\left(N-\frac{NB_1}{B_1+B_2}\right)P\left(X\le\frac{NB_1}{B_1+B_2}\right)\\
=&\frac{N}{B_1+B_2}+\frac{1}{B_1}E\left(\left(X-\frac{NB_1}{B_1+B_2}\right)I\left\{X\ge\frac{NB_1}{B_1+B_2}\right\}\right)\\
&+\frac{1}{B_2}E\left(\left(\frac{NB_1}{B_1+B_2}-X\right)I\left\{X\le\frac{NB_1}{(B_1+B_2)}\right\}\right), \qquad (13)
\end{aligned}
$$

where $I\{A\}$ is the indicator function of the set A. Let Φ, Φ^c stand for the standard Normal distribution function and its complement. Let ϕ denote the standard normal density function. Let $\alpha = NB_1/(B_1+B_2)$, $\mu = Np$, and $\sigma = \sqrt{Np(1-p)}$. Then the right hand side of this equation can be simplified as:

$$
\begin{aligned}
&\frac{N}{B_1+B_2}+\frac{1}{\sigma B_1}\int_{\frac{NB_1}{B_1+B_2}}^{\infty}\left(X-\frac{NB_1}{B_1+B_2}\right)\phi\left(\frac{X-\mu}{\sigma}\right)dX\\
&\qquad+\frac{1}{\sigma B_2}\int_{-\infty}^{\frac{NB_1}{B_1+B_2}}\left(\frac{NB_1}{B_1+B_2}-X\right)\phi\left(\frac{X-\mu}{\sigma}\right)dX=\\
=&\frac{N}{B_1+B_2}+\frac{1}{\sigma B_1}\int_{\frac{NB_1}{B_1+B_2}}^{\infty}(X-\mu)\phi\left(\frac{X-\mu}{\sigma}\right)dX\\
&-\frac{1}{\sigma B_2}\int_{-\infty}^{\frac{NB_1}{B_1+B_2}}(X-\mu)\phi\left(\frac{(X-\mu)}{\sigma}\right)dX\\
&+\frac{1}{B_1}\left(\mu-\frac{NB_1}{B_1+B_2}\right)\Phi^c\left(\frac{\alpha-\mu}{\sigma}\right)\\
&-\frac{1}{B_2}\left(\mu-\frac{NB_1}{B_1+B_2}\right)\Phi\left(\frac{\alpha-\mu}{\sigma}\right)\\
=&\frac{N}{B_1+B_2}+\frac{1}{B_1}\left(\mu-\frac{NB_1}{B_1+B_2}\right)\Phi^c\left(\frac{\alpha-\mu}{\sigma}\right)\\
&-\frac{1}{B_2}\left(\mu-\frac{NB_1}{B_1+B_2}\right)\Phi\left(\frac{\alpha-\mu}{\sigma}\right)\\
&-\frac{\sigma}{B_1}\int_{\frac{NB_1}{B_1+B_2}}^{\infty}d\phi\left(\frac{X-\mu}{\sigma}\right)+\frac{\sigma}{B_2}\int_{-\infty}^{\frac{NB_1}{B_1+B_2}}d\phi\left(\frac{X-\mu}{\sigma}\right)
\end{aligned}
$$

$$= \frac{N}{B_1+B_2} + (\mu-\alpha)\left(-\frac{1}{B_2}\Phi\left(\frac{\alpha-\mu}{\sigma}\right)+\frac{1}{B_1}\Phi^c\left(\frac{\alpha-\mu}{\sigma}\right)\right)$$
$$+\sigma\phi\left(\frac{\alpha-\mu}{\sigma}\right)\left(\frac{1}{B_2}+\frac{1}{B_1}\right) \tag{14}$$

Proof of Proposition 1: This expression given in equation (1) can be minimized by using a search procedure. A approximate but quicker result can be obtained by treating σ as independent of p (this is certainly not true, but this approximation turns out to yield very good estimates for p^*). Then the expression to be minimized can be written as:

$$\begin{aligned} H(z) &= -\frac{\alpha-\mu}{\sigma}\left(-\frac{1}{B_2}\Phi\left(\frac{\alpha-\mu}{\sigma}\right)+\frac{1}{B_1}\Phi^c\left(\frac{\alpha-\mu}{\sigma}\right)\right) \\ &\quad +\phi\left(\frac{\alpha-\mu}{\sigma}\right)\left(\frac{1}{B_2}+\frac{1}{B_1}\right) \qquad (15) \\ &= -z\left(-\frac{1}{B_2}\Phi(z)+\frac{1}{B_1}\Phi^c(z)\right)+\phi(z)\left(\frac{1}{B_2}+\frac{1}{B_1}\right) \qquad (16) \end{aligned}$$

where z is the standard Normal deviate,

$$\frac{dH(z)}{dz} = \frac{1}{B_2}\Phi(z) - \frac{1}{B_1}\Phi^c(z), \tag{17}$$

and

$$\frac{d^2H(z)}{dz^2} = \frac{1}{B_2}\phi(z) + \frac{1}{B_1}\phi(z) > 0. \tag{18}$$

Therefore $H(z)$ is minimized by setting the first derivative equal to zero, giving

$$\frac{1}{B_2} = \Phi^c(z)\left(\frac{1}{B_1}+\frac{1}{B_2}\right) \Leftrightarrow \Phi^c\left(\frac{\alpha-Np}{\sqrt{Np(1-p)}}\right) = \frac{B_1}{B_1+B_2}. \tag{19}$$

References

[1] D. Bitton and J. Gray. Disk Shadowing. *Proceedings of the 14th International Conference on Very Large Data Bases*, pp. 331-338, August, 1988.

[2] J. A. Buzacott and J. G. Shanthikumar. *Stochastic Models of Manufacturing Systems*, Prentice Hall. NJ, 1993.

[3] L. T. Chen and D. Rotem. Declustering Objects for Visualization. *Proceedings of the 19th International Conference on Very Large Data Bases*, pp. 85-96, 1993.

[4] L. T. Chen, D. Rotem and S. Seshadri. Declustering databases on heterogeneous disk systems. *Proceedings of the 21th International Conference on Very Large Data Bases*, pp. 110-121, 1995.

[5] D. J. DeWitt and S. Ghandeharizadeh. Hybrid-Range Partitioning Strategy: A New Declustering Strategy for Multiprocessor Database Machine. *Proceedings of the 16th International Conference on Very Large Data Bases*, 481-492, August, 1990.

[6] H. C. Du. Disk Allocation Methods for Binary Cartesian Product Files. *BIT*, 26, pp. 138-147, 1986.

[7] C. Faloutsos and P. Bhagwat. Declustering Using Fractals. *Second International Conference on Parallel and Distributed Computing (PDIS)"*, pp. 18-25, January, 1992.

[8] F. Faloutsos and D. Metaxas. Disk Allocation Methods Using Error Correcting Codes. *IEEE Transactions on Computers*, **40**(8), pp. 907-914, August, 1991.

[9] S. Ghandeharizadeh, I. Ramos, Z. Asad and W. Qureshi. Object Placement in Parallel HyperMedia Systems. *Proceedings of the 17th international Conference on Very Large Data Bases*, pp. 243-254, September, 1991.

[10] R. Lo and N. S. Matloff. A Probabilistic Limit on the Virtual Size of Replicated Disk Systems. *IEEE Transactions on Knowledge Engineering"*, **4**(1), pp. 862-871, 1992.

[11] C. A. Polyzois, A. Bhide and D. Dias. Disk Mirroring with Alternating Deferred Updates. *Proceedings of the 19th International Conference on Very Large Data Bases*, pp. 604-617, August, 1993.

[12] D. Rotem and G. Schloss. I/O Performance of Fully-Replicated Disk Systems. *Proceedings of the Second Workshop on the Management of Replicated Data*, pp. 68-73, November, 1992.