Replicated Parallel I/O without Additional Scheduling Costs[⋆]

Mikhail Atallah and Keith Frikken

Purdue University

Abstract. A common technique for improving performance in a database is to decluster the database among multiple disks so that data retrieval can be parallelized. In this paper we focus on answering range queries in a multidimensional database (such as a GIS), where each of its dimensions is divided uniformly to obtain tiles which are placed on different disks; there has been a significant amount of research for this problem (a subset of which is [1,2,3,4,5,6,7,8,9,11,12,13,14,15]). A declustering scheme would be optimal if any range query could be answered by doing no more than $\lceil$# of tiles inside the range/# of disks $\rceil$ retrievals from any one disk. However, it was shown in [1] that this is not achievable in many cases even for two dimensions, and therefore much of the research in this area has focused on developing schemes that performed close to optimal. Recently, the idea of using replication (i.e. placing records on more than one disk) to increase performance has been introduced. [7, 12,13,15]. If replication is used, a retrieval schedule (i.e. which disk to retrieve each tile from) must be computed whenever a query is being processed. In this paper we introduce a class of replicated schemes where the retrieval schedule can be computed in time O(# of tiles inside the query's range), which is asymptotically equivalent to query retrieval for the non-replicated case. Furthermore, this class of schemes has a strong performance advantage over non-replicated schemes, and several schemes are introduced that are either optimal or are optimal plus a constant additive factor. Also presented in this paper is a strictly optimal scheme for any number of colors that requires the lowest known level of replication of any such scheme.

1 Introduction

A typical bottleneck in many systems is I/O; to reduce the effect of this bottleneck data can be declustered onto multiple disks to facilitate parallel retrieval of the data. In a multi-dimensional database, such as a GIS or a spatio-temporal database, the dimensions can be tiled uniformly to form a grid, and when answering a range query in such a system, only the tiles that contain part of the

⋆ Portions of this work were supported by Grants EIA-9903545 and ISS-0219560 from the National Science Foundation, Contract N00014-02-1-0364 from the Office of Naval Research, by sponsors of the Center for Education and Research in Information Assurance and Security, by Purdue Discovery Park's e-enterprise Center, and by the GAANN fellowship.

query need to be retrieved. In such an environment, a declustering scheme attempts to place the tiles on disks in such a way that the average range query is answered as efficiently as possible. If the database is treated like a grid and the disks as colors, then this can be stated as a grid coloring problem. For the rest of the paper we use record and tile synonymously and likewise use declustering and coloring interchangeably.

Given a database declustered on k disks and a range query Q contained on m tiles, Q is answered optimally if no more than $\lceil \frac{m}{k} \rceil$ tiles are retrieved from any one disk. A declustering is called strictly optimal if all range queries can be answered optimally, however it was shown in [1] that this is not achievable except in a few limited circumstances. Thus there has been a significant amount of work to develop declustering schemes that have close to optimal performance, a sampling of which are in [2,3,4,5,6,9,11,14].

To improve performance further the idea of using replication (i.e. placing each tile on multiple disks) has been introduced [7,12,13,15]. When replication is used each tile in a query can be retrieved from multiple places which allows greater flexibility when answering the query. In order to use replication an algorithm for computing an optimal retrieval schedule is required (i.e. which disk do you retrieve each tile from). Algorithms for computing this schedule are given in [7, 12,13,15]. The most general of these runs in time $O(rm^2 + mk)$ where r is the most number of disks that a tile is stored on, m is the number of tiles to be retrieved, and k is the number of disks. (For more information on work using replication see Section 2). One problem with replication is that it adds a non-negligible overhead to query response. In this paper we define a class of coloring techniques, which we call the grouping schemes, for which a schedule of retrievals can be computed in time $O(\#$ of tiles to be retrieved) (from here on we refer to this as $O(\#$ of tiles)), which is asymptotically equivalent to the time required to compute a schedule for a non-replicated scheme. This technique essentially transforms existing coloring schemes into replicated schemes by placing disks into groups and placing tiles on all disks in a group; when the group size is two this is equivalent to RAID level 1. Previously, the only general strict optimal solution for any number of disks was the Complete Coloring [7,15], which places each tile on every disk. We introduce several new schemes that are either have strictly optimal performance for all queries or will answer any query in no more time than a strictly optimal scheme plus an additive constant; these new schemes have the lowest known level of replication for such performance bounds. Furthermore, these grouping schemes are shown to have stronger experimental performance than schemes without replication.

The outline of this paper is as follows: Section 2 discusses previous work in this area, in Section 3 the grouping schemes are introduced and schemes that achieve strictly optimal performance or are a constant additive factor above an optimal solution, Section 4 contains experimental data showing the performance of the grouping schemes, and Section 5 concludes the paper.

2 Related Work

Given an n-dimensional database with each dimension divided uniformly to form tiles, if tiles are placed on different disks, then the retrieval of records during query processing can be parallelized. The I/O time in such a system is time that it takes to retrieve the maximum number of tiles stored on the same disk. The problem of placing the records so that the response times for range queries is minimized has been well studied; this section presents a survey of this work.

Given a database declustered onto k disks and a range query Q contained on m tiles, an optimal tile declustering would require no more than $\lceil \frac{m}{k} \rceil$ retrievals from any one disk. It was shown in [1] that this bound is unachievable for all range queries in a grid except in a few limited circumstances. Since there are many cases where no scheme can achieve this optimal bound, several schemes have been developed to achieve performance that is close to optimal. To quantify "close to optimal", define the additive error of a declustering scheme to be the maximum over all range queries Q of the value $(rettime(Q) - \lceil \frac{m}{k} \rceil)$, where $rettime(Q)$ is defined as the retrieval time for query Q (i.e. it is the maximum number of tiles in Q retrieved from a single disk). These schemes include Disk Modulo DM [6], Fieldwise eXclusive (FX) or [9], the cyclic schemes (including RPHM, GFIB, and EXH) [11], GRS [4], a technique developed by Atallah and Prabhakar [2] which we will call RFX, and several techniques based on discrepancy theory [5,14] (for an introduction to discrepancy theory see [10]). Note that these are just a subset of the declustering techniques that have been developed for this problem.

Suppose we are given k colors. The DM approach [6] assigns tile (x, y) to $(x + y) \bmod k$. The FX approach [9] assigns tile (x, y) to $(x \oplus y) \bmod k$. Cyclic allocation schemes [11] choose a skip value s such that $gcd(k, s) = 1$ and assigns tile (x, y) to $(x + sy) \bmod k$. The choice of the skip value, s, is what defines the scheme. In RPHM (Relatively Prime Half Modulo), s is defined to be the integer nearest to $\frac{k}{2}$ that is relatively prime to k. The GFIB (Generalized FIBonacci) scheme defines s to be an approximate of the previous Fibonacci number (by using the closed formula) that is relatively prime to k. The EXH (Exhaustive) scheme takes all values of s where $gcd(s, k) = 1$ and finds the one that optimizes a certain criterion, for example minimizing the additive error is a possible criterion. Another class of schemes are the permutation schemes [4], in these schemes a permutation ϕ of the numbers in $\{0, ..., k-1\}$ is chosen and then tile (x, y) is assigned color $(x - \phi^{-1}((y) \bmod k))$. Examples of permutation schemes are DM, the cyclic schemes, and GRS. In the GRS scheme [4] the permutation is computed as follows: i) $\forall i \in \{0, ..., k-1\}$ compute the fractional part of $\frac{2i}{1+\sqrt{5}}$, and call it k_i and then ii) sort the values k_i and use this to define the permutation. A scheme based on the Corput set is defined in [14] that is similar to GRS except that the k_i values are $\frac{a_0}{2} + \frac{a_1}{4} + \frac{a_2}{8} + ... + \frac{a_{k-1}}{2^k}$ where $a_{k-1}...a_1 a_0$ is the binary representation of i. In [2], the RFX scheme was presented that was later found in [3] to be equivalent to $(x \oplus y^R) \bmod k$, where y^R is the $(\lceil \log k \rceil)$-bit reversal of y. For brevity, the details of higher dimensional schemes are not provided.

It was shown in [14] that the additive error for k colors in two dimensions is $\Omega(\log k)$, and that in $d(\geq 3)$ dimensions it is $\Omega(\log^{\frac{d-1}{2}} k)$. In two dimensions, schemes have been developed (RFX, GRS, and schemes based in discrepancy theory [2,14]) that have a provable upper bound of $O(\log k)$ on additive error. For higher dimensions $d(\geq 3)$, two schemes are given in [5] with additive error $O(\log^{(d-1)} k)$, which are the schemes with the lowest proven asymptotic bound on additive error. A recent trend has been to use replication [7,12,13,15] to increase performance further. Several query scheduling algorithms have been given in previous work, but the only general algorithm that works for any type of replication is in [7] and runs in time $O(rm^2+mk)$ where r is the most number of disks that a tile is stored on, m is the number of tiles to be retrieved, and k is the number of disks. In [12,13] it was proven that if tiles are stored on two random disks then the probability of requiring more than ($\lceil$ (# of tiles/# of disks) $\rceil$ +1) retrievals from a single disk for a random query approaches 0 as the number of disks gets large. In [15] replication was used to achieve optimal solutions for up to 15 disks. A strictly optimal scheme, called Complete Coloring (CC), for any number of disks by storing all tiles on all disks was introduced in [7,15]. The SRCDM scheme was introduced in [7], and has an additive error no larger than 1, but requires the number of disks be a perfect square (n^2) and requires that each tile is placed on n disks.

3 Grouping Replication Scheme

In this section the grouping schemes are introduced. Section 3.1 defines some notations that will be needed before defining this class of schemes. In Section 3.2, the grouping schemes are defined along with an algorithm that computes the retrieval schedule in time $O(\#\text{of tiles})$. Section 3.3 contains several schemes that have an additive error that is 0 or is $O(1)$. Finally, in Section 3.4 we provide a strictly optimal coloring scheme that works for any number of colors.

3.1 Notations and Terminology

Before we can formally define the grouping schemes we need to define some notation and terminology. A *non-replicated coloring function* C for d dimensions and m disks is a function $C : \aleph^d \rightarrow \{0, ..., m-1\}$, essentially C maps a tile to a disk. A *replicated coloring function* C with *level of replication* r for d dimensions and m disks is a function $C : \aleph^d \rightarrow \bigcup_{i=1}^{r} \{0, ..., m-1\}^i$, essentially C maps a tile to the set of disks (with size no more than r) that contain the tile. Since the replicated coloring function is a generalization of the non-replicated coloring function, we assume all coloring functions are replicated for the rest of the paper. A convenient shorthand notation for coloring schemes is (C, m, r, d) which states the coloring function C declusters a d dimensional grid onto m disks with a level of replication r. Two coloring schemes (C, m, r, d) and (D, m, r, d) are said to be *equivalent* if and only if there is a bijection $f : \{0, ..., m-1\} \rightarrow \{0, ..., m-1\}$ such that $i \in C(x_1, ..., x_d)$ iff $f(i) \in D(x_1, ..., x_d)$. Essentially schemes are equivalent if there is a rearrangement of the colors that will make them identical, and it is obvious that equivalent schemes have identical retrieval time for any query.

3.2 Definition of Grouping Schemes

A scheme is considered to be *formed with groups* if the colors are partitioned into sets and tiles are assigned to partitions where assigning a partition to a tile is equivalent to placing it on all disks in that partition. The motivation for this class of schemes is to be able to distribute the additive error of the coloring that assigns tiles to partitions among the different members of the partition. Hence, the additive error of any one member of the partition will be smaller, and thus reducing the additive error of the scheme.

Formally, a coloring scheme (C, m, r, d) is considered formed by groups if the colors can be partitioned into sets $S_1, S_2, ..., S_k$ with at least one set where $|S_i| > 1$ such that if $C(x_1, x_2, ..., x_d) = S$ and the following holds: if $(S_i \cap S) \neq \emptyset$, then $S_i \subseteq S$. Such a scheme is called *simple* if the last constraint is changed to: if $(S_i \cap S) \neq \emptyset$, then $S_i = S$. A scheme formed by groups is said to have *equal partitions* if each partition S_i is identical in size, or equivalently $|S_i| = \frac{m}{k}$ for all i.

It is possible to transform any coloring scheme (C, m, r_1, d) into a scheme formed by groups with equal partitions (of size r_2) (C', mr_2, r_1r_2, d), where C' is defined as: $C'(x_1, ..., x_d) = \bigcup_{s \in C(x_1,...,x_d)} \{im + s | 0 \leq i < r_2\}$, we denote this transformation process by $GROUP((C, m, r_1, d), r_2)$ ((C, m, r_1, d) is referred to as the *base scheme* in what follows). A scheme defined with $GROUP$ is simple if $r_1 = 1$. Now any scheme defined with $GROUP$ is a scheme formed by groups with equal partitions, but any scheme formed by groups with equal partitions is equivalent to a scheme that can be defined with $GROUP$ (proof omitted). We call the set of schemes defined by $GROUP$ the *grouping schemes*.

There have been scheduling algorithms defined for any replicated algorithm that will work for any replicated scheme, but for simple grouping schemes (represented by $GROUP((C, m, 1, d), r)$) there is a scheduling algorithm that runs in time $O(m)$ and executes with one pass over the tiles (see **RetrieveTiles** below). The algorithm uses a function **SetSchedule**(*tile*,*disk*) which sets the schedule to retrieve *tile* from *disk*.

```
begin RetrieveTiles(Query, $(C, mr, r, d) = GROUP((D, m, 1, d), r)$)
    $A[]$ := array initialized to 0 of size $m$.
    forall $t = (t_1, ..., t_d)$ in Query do
        $c := C(t)$
        SetSchedule($t$,$c + A[c]$)
        $A[c] := ((A[c] + m) \bmod (mr))$
    endfor
end RetrieveTiles
```

Thus simple grouping schemes can be used without having to incur the additional scheduling costs of other replicated schemes. In addition to this, the additive error of a grouping scheme with level of replication l is bounded by the $\lceil \frac{o}{l} \rceil$ where o is the additive error of the base scheme (see Theorem 3-3). Before this can be proven we need Lemmas 3-1 and 3-2.

Lemma 3-1: If the coloring scheme (C, m, r, d) has an additive error of o, then $GROUP((C, m, lr, d), l)$ will have a response time no larger than $\lceil \frac{\lceil \frac{x}{k} \rceil + o}{l} \rceil$ for x records.

Proof: For the x records the original coloring scheme will have at most $(\lceil \frac{x}{k} \rceil + o)$ instances of any one color which means there will be at most $(\lceil \frac{x}{k} \rceil + o)$ instances of any group. These values can be distributed equally among the l colors in that group to obtain a maximum response time of $\lceil \frac{\lceil \frac{x}{k} \rceil + o}{l} \rceil$. **QED**

Lemma 3-2: $\lceil \frac{\lceil \frac{x}{k} \rceil + o}{l} \rceil \leq \lceil \frac{x}{kl} \rceil + \lceil \frac{o}{l} \rceil$

Proof: Let $x = a(kl) + bk + c$, where $0 \leq b < l$ and $0 \leq c < k$. There are two cases to consider: $(b = 0$ and $c = 0)$ or $(b \neq 0$ or $c \neq 0)$.

Case 1: $(b = 0$ and $c = 0)$: $\lceil \frac{\lceil \frac{x}{k} \rceil + o}{l} \rceil = \lceil \frac{\lceil \frac{a(kl)}{k} \rceil + o}{l} \rceil = \lceil \frac{al + o}{l} \rceil = a + \lceil \frac{o}{l} \rceil = \lceil \frac{a(kl)}{kl} \rceil + \lceil \frac{o}{l} \rceil = \lceil \frac{x}{kl} \rceil + \lceil \frac{o}{l} \rceil$

Case 2: $(b \neq 0$ or $c \neq 0)$: $\lceil \frac{\lceil \frac{x}{k} \rceil + o}{l} \rceil = \lceil \frac{\lceil \frac{akl + bk + c}{k} \rceil + o}{l} \rceil = \lceil \frac{al + b + \lceil \frac{c}{k} \rceil + o}{l} \rceil \leq a + \lceil \frac{b + 1 + o}{l} \rceil \leq a + \lceil \frac{l + o}{l} \rceil = a + 1 + \lceil \frac{o}{l} \rceil = \lceil \frac{a(kl) + bk + c}{kl} \rceil + \lceil \frac{o}{l} \rceil = \lceil \frac{x}{kl} \rceil + \lceil \frac{o}{l} \rceil$.

In either case $\lceil \frac{\lceil \frac{x}{k} \rceil + o}{l} \rceil \leq \lceil \frac{x}{kl} \rceil + \lceil \frac{o}{l} \rceil$. **QED**

Theorem 3-3: If the coloring scheme (C, m, r', d) has an additive error of o, then $GROUP((C, m, r', d), r)$ will have an additive error no larger than $\lceil \frac{o}{r} \rceil$.

Proof: Follows directly from Lemma 3-1 and Lemma 3-2. **QED**

This last theorem implies that the additive error for a coloring scheme can be reduced by using this grouping method. Since the additive error can be reduced the expected value above optimal will also be reduced. To summarize this section, a class of replicated schemes can be defined with the $GROUP$ transformation, which we call the grouping schemes. A subset of these schemes are simple and for this subset there are two significant benefits compared to non-replicated schemes including: i) queries can be processed in time proportional to the number of records which is asymptotically optimal, and ii) there is a performance increase.

3.3 Achieving Optimal and Constant Additive Error

In this section schemes with 0 and $O(1)$ additive error are introduced.

Corollary 3-4: If a scheme (C, m, r_1, d) is strictly optimal so is $GROUP((C, m, r_1, d), r_2)$.

Proof: Since (C, m, r_1, d) is strictly optimal the additive error will be 0, and thus by Theorem 3-3, the additive error of $GROUP((C, m, r_1, d), r_2)$ will also be 0, and thus is strictly optimal. **QED**

The previous corollary implies that any scheme with level of replication r that is optimal for c disks can be transformed using $GROUP$ into a scheme that is optimal for ck disks with level of replication kr. It is possible to color a two dimensional grid optimally with 1, 2, or 3 (or 5 in 2-D) colors using RPHM in two dimensions and DM in higher dimensions. Hence, it is possible to color a grid with r, $2r$, or $3r$, (or $5r$ in 2-D) with level of replication r in a strictly optimal fashion (by Corollary 3-4). Furthermore, these schemes are simple and so **RetrieveTiles** can be used to retrieve the tiles in time proportional to the number of records. The CC scheme described in [7,15] is the scheme defined above that uses a base scheme with only 1 color. In addition to these optimal schemes there are grouping schemes that achieve an additive error that is $O(1)$.

Corollary 3-5: If a scheme (C, m, r_1, d) has an additive error of $O(f(m))$ for some function f then is $GROUP((C, m, r_1, d), x)$, where $x > f(m)$ has an additive error that is $O(1)$.

Proof: Since the scheme (C, m, r_1, d) has an additive error of $O(f(m))$, then the additive error is bounded by $af(m)$ for some constant a. Now by theorem 3-3, $GROUP((C, m, r_1, d), x)$ will have an additive error no larger than $\lceil \frac{af(m)}{x} \rceil \leq a$ and thus is $O(1)$. **QED**

The following is a table of grouping schemes with base schemes with m colors that have an additive error which is $O(1)$ but can be scheduled with **RetrieveTiles** (it is assumed m is a power of 2 for the RFX scheme):

Base Scheme	**Dimensions**	**Level of Replication**	**Additive Error**
LHDM [8]	d	$(m-1)^{d-1} - 1$	1
RFX, GRS, and other schemes [2,3,14]	2	$\log m$	$O(1)$
RFX [2,3]	2	$2 \log m - 3$	1
Schemes in [5]	d	$\log^{d-1}(m)$	$O(1)$

3.4 Generalizing Optimal Additive Error

In the previous section schemes were introduced that were strictly optimal, but these schemes are applied in the situation where the number of colors was a multiple of the number of colors in a base scheme that is optimal (i.e. 1, 2, or 3 (or 5 in 2-D)). The CC coloring is an instance of the previous scheme and is strictly optimal for any number of colors, but it requires that the level of replication be the number of colors, which may be unreasonable for many applications. In this section a strictly optimal scheme for any number of colors is given with a level of replication close to half the number of colors.

The scheme presented here is a generalization of $GROUP((C, 2, 1, d), k)$ where C is the DM coloring scheme that is strictly optimal for any number of colors. In the case, where the number of colors is even, we are trivially done using schemes discussed in the previous section. Suppose the number of colors is odd (i.e. $m = 2k + 1$), to create a scheme for m colors place $2k$ of the colors using the grouped DM scheme with level of replication k and then place the entire database on the last disk. Note that the level of replication for such a scheme for m disks is $\lfloor \frac{m}{2} \rfloor + (m \bmod 2)$ which is about $\frac{m}{2}$. It can be proven that this scheme is strictly optimal, but we omit this proof due to space constraints.

A possible criticism of this scheme is that if you can place the entire database on a single disk, then why not use the CC mechanism for simplicity. However, in this case it is only required that a single disk be large enough to hold the entire database. This may not be reasonable for large databases, but is reasonable in some situations.

This generalized approach can be extended to grouping schemes with base schemes with 3 (and 5 in 2-D) disks in a similar fashion (we omit the details due to space constraints). It is not true however that if you have an optimal scheme for k colors that if you put all tiles on another disk that the solution will be optimal for $k+1$ disks.

The scheme defined in this section constitutes a general strictly optimal schemes with the lowest known level of replication. With some modification to our scheduling algorithm the schedule can be computed in time $O(\#$ of records$)$. We give a verbal description of the algorithm here for when there are $2k+1$ colors and the scheme described above is used. Essentially there are two groups of k colors and 1 extra color. We know that an optimal schedule is achievable so we determine what optimal is, and call it o. Assign up to the first ko tiles of each group to disks in that group, such that no more than o tiles are assigned to any one disk, and if there are any tiles remaining after this has been done to both groups assign these leftovers to the last disk which is not in either group.

4 Experiments

For this section, experiments were performed to compare the performance between grouped schemes with level of replication as 2 and non-grouped schemes. The comparison criterion that is used is the expected deviation from optimal for all queries. To compute the expected deviation from optimal for a coloring scheme (C, m, r, d) we compute the expected value from optimal of all wraparound queries in an d dimensional grid with side lengths equal to m. There is a finite number of queries in such a grid so this value can be computed exactly for smaller m values, but is estimated for larger values. This estimation is done by taking a random sampling of queries and computing the expected deviation from optimal of these queries. It is worth noting that when an exact value is computed that the maximum additive error found will be the maximum additive error in any grid (see [8], which can be generalized to grouping schemes, but this generalization is omitted). To perform the comparison between the replicated and non-replicated schemes we use a hybrid coloring. Given a set of colorings this hybrid coloring uses the coloring that minimizes the expected deviation from optimal for a specific number of disks, i.e. the hybrid coloring uses the best coloring in the set for a specific situation. The comparison is figure 1 is between the hybrid coloring of a set of non-replicated schemes and the hybrid coloring for these schemes transformed with $GROUP$ using level of replication of 2.

The set of non-replicated coloring schemes used are DM [6], FX (for powers of 2) [9], RFX (for powers of 2) [2], RPHM [11], GFIB [11], GRS [4], and a scheme

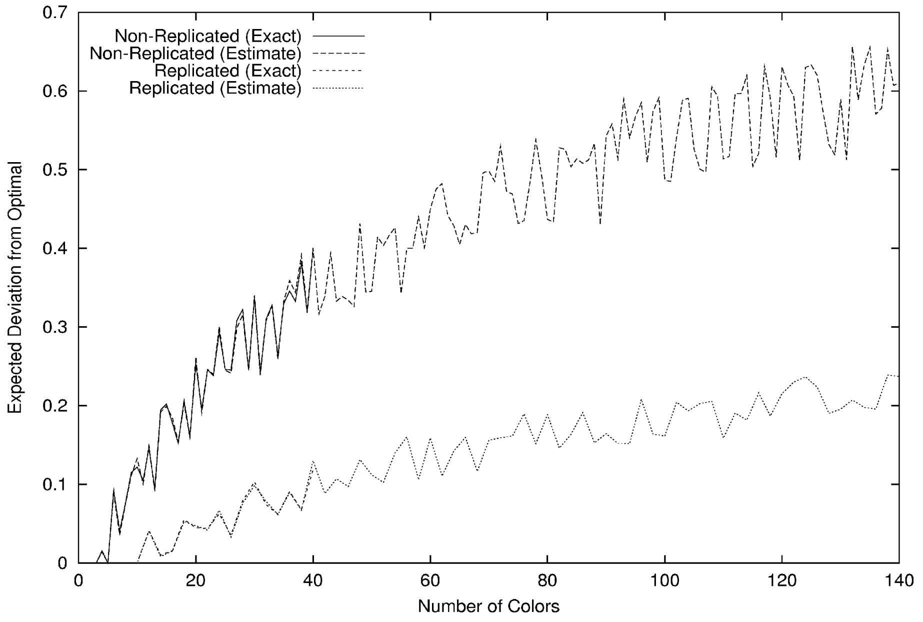

Fig. 1. Expected Deviation from Optimal for 2-D Schemes

based on the Corput set [14]. For the grouping schemes we use the grouped version of these schemes with level of replication 2. When the number of disks is no more than 40, exact values were computed, but estimates were used for up to 140 disks. These estimates were made by looking at 5000 queries (chosen randomly with uniform distribution) in the grid using the mean deviation as the estimate. The results are displayed in Figure 1.

Figure 1 is interesting for several reasons. First, it shows that the estimate is accurate for predicting the expected deviation for values up to 40. Also, it shows that the grouping schemes perform far better than the non-replicated schemes, since the expected deviation from optimal is 2-3 times larger for non-replicated colorings than for grouping schemes. Thus if a replication level of 2 is used, then range query performance will be improved substantially.

5 Conclusions

When declustering data, there are three inhibiting factors that may prevent the usage of replication: i) there is not enough disk space on each disk to contain enough information, ii) the slow down that occurs with query scheduling for replication is too overwhelming, and iii) the benefit from replication is not significant. We have introduced a class of schemes, called the grouping schemes, which eliminate conditions (ii) and (iii). Condition (ii) is eliminated because the grouping schemes can be scheduled in time O(number of tiles), and it was shown in Section 4, that these techniques perform extremely well even if the level of replication is 2 which eliminates condition (iii). Thus an important conclusion about the usage of replication can be stated: If there is enough room on the disks to facilitate replication, then replication should be used. Furthermore, a strictly optimal scheme for any number of colors with the lowest known level of replication for such a solution was presented along with several schemes with additive error that is O(1) were given that have fewer restrictions on the number of disks and have a lower level of replication than previous schemes that achieve an $O(1)$ bound on additive error (the authors know of only one such previous scheme, which is SRCDM).

References

1. K. Abdel-Ghaffar and A. E. Abbadi. Optimal allocation of two-dimensional data. In *International Conference on Database Theory*, pages 409–418, 1997.
2. M. J. Atallah and S. Prabhakar. (almost) optimal parallel block access to range queries. In *Proceedings of the nineteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 205–215. ACM Press, 2000.
3. R. Bhatia, R. Sinha, and C.-M. Chen. Hierarchical declustering schemes for range queries. In *In 7th Int'l Conf. on Extending Database Technology*, 2000.
4. R. Bhatia, R. K. Sinha, and C.-M. Chen. Declustering using golden ratio sequences. In *ICDE*, pages 271–280, 2000.
5. C.-M. Chen and C. T. Cheng. From discrepancy to declustering: near-optimal multidimensional declustering strategies for range queries. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 29–38. ACM Press, 2002.
6. H. Du and J. Sobolewski. Disk allocation for cartesian product files on multiple disk systems. *ACM Transactions on Database System*, pages 82–101, 1982.
7. K. Frikken, M. Atallah, S. Prabhakar, and R. Safavi-Naini. Optimal parallel i/o for range queries through replication. In *Proceedings of 13th Intl. Conf. on Database and Expert Systems Application (LNCS 2453)*, pages 669–678.
8. B. Himatsingka, J. Srivastava, J.-Z. Li, and D. Rotem. Latin hypercubes: A class of multidimensional declustering techniques, 1994.
9. M. H. Kim and S. Pramanik. Optimal file distribution for partial match retrieval. In *Proceedings of the 1988 ACM SIGMOD international conference on Management of data*, pages 173–182. ACM Press, 1988.
10. J. Matousek. *Geometric discrepancy, an illustrated guide.* Springer-Verlag, 1999.
11. S. Prabhakar, K. Abdel-Ghaffar, D. Agrawal, and A. E. Abbadi. Cyclic allocation of two-dimensional data. Technical Report TRCS97-08, 1, 1997.
12. P. Sanders. Reconciling simplicity and realism in parallel disk models. In *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, pages 67–76. ACM Press, 2001.
13. P. Sanders, S. Egner, and J. Korst. Fast concurrent access to parallel disks. In *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, pages 849–858. ACM Press, 2000.
14. R. K. Sinha, R. Bhatia, and C.-M. Chen. Asymptotically optimal declustering schemes for range queries. In *International Conference on Database Theory*, 2001.
15. A. Tosun and H. Ferhatosmanoglu. Optimal parallel i/o using replication. Technical Report OSU-CISRC-11/01-TR26, 2001.