CENTER FOR
PARALLEL OPTIMIZATION

OPTIMAL BALANCED ASSIGNMENTS AND A PARALLEL DATABASE APPLICATION

by

Shahram Ghandeharizadeh
Robert R. Meyer
Gary L. Schultz
Jonathan Yackel

Computer Sciences Technical Report #986

December 1990

Optimal Balanced Assignments and a Parallel Database Application

Shahram Ghandeharizadeh* Robert R. Meyer Gary L. Schultz
Jonathan Yackel

Abstract

In parallel database systems, distribution of the data among the processors has a significant impact on the response time and throughput of the system. The benefits of parallelism (using multiple processors to execute a query) must be balanced against its costs (communication, startup, and termination overhead). We formalize the problem of minimizing overhead while partitioning data uniformly across the processors. We derive lower bounds on these combinatorial problems and demonstrate how processors may be optimally assigned so as to achieve these lower bounds for a number of problem classes.

Acknowledgments

This research was partially supported by the Air Force Office of Scientific Research under grant 89-0410, the Defense Advanced Research Projects Agency under contract N00039-86-C-0578, and by the National Science Foundation under grants CCR-8709952 and DCR-8512862.

1 Introduction

We first consider the application that motivated for the general combinatorial optimization problems developed below.

1.1 A Parallel Database Application

In highly-parallel database machines (e.g., Gamma [7], Bubba [5], DBC/1012 [18], Non-Stop SQL [17], XPRS [16] and Volcano [10]) relations are horizontally partitioned across multiple processors. (Livny *et al* [14] and Ries and Epstein [15] introduced the concept of horizontal partitioning.) This allows each processor to execute a portion of a query in parallel with the other processors, resulting in a lower response time for the query. In these systems, the number of processors assigned to a relation determines the degree of parallelism for a query accessing that relation.

*Present affiliation: Computer Science Department, University of Southern California.

Social_Security

Salary	0-9	10-19	20-29	30-39	40-49	50-inf
0 - 20	0	0	3	3	6	6
21 - 45	0	0	3	3	6	6
46 - 55	1	1	4	4	7	7
56 - 60	1	1	4	4	7	7
61 - 80	2	2	5	5	8	8
81-inf	2	2	5	5	8	8

Figure 1: A 2-dimensional grid on the EMP relation

However, there is communication overhead associated with initiating and terminating a query on multiple processors. Furthermore, this overhead increases as a function of the number of processors used to execute a query [1]. In order to minimize overhead while balancing the workload among the processors, Multi-Attribute GrId deClustering (MAGIC) introduced by Ghandeharizadeh [9] partitions a relation by assigning ranges of several attribute values to each processor in the system. To illustrate MAGIC declustering, consider the Employee relation: `EMP (Social_Security, Name, Salary, Age, Dept)`. For a parallel system consisting of 9 processors, MAGIC partitions the EMP relation by establishing ranges of Salary and Social_Security attribute values as shown in figure 1. Each cell of this grid corresponds to a fragment of the relation and must be assigned to some processor. For example, the cell which contains records with Salary attribute values that range from 46 to 55 and Social_Security attribute values that range from 0 to 9 is assigned to processor 1.

Given a query on either the Social_Security or Salary attribute, the predicate of the query maps to either a row or a column (termed a "slice") of the grid and the corresponding three processors are used to execute it. For example, if a query retrieves the record that corresponds to the employee with a Social_Security attribute value five, the value of the query predicate maps to the first column of the grid and processors 0, 1, and 2 are used to execute it. Note that, for the assignment depicted by figure 1, every such query requires 3 processors and every processor is assigned 4 cells. (The grid in figure 1 is produced by a "blocking" procedure, discussed in §4, that constitutes an optimal assignment in some cases including the one above.)

[1] This overhead is primarily in the form of additional messages to control the execution of the query on additional processors and, in the Gamma database machine [7], increases linearly with the number of employed processors.

Although we concentrate on the limiting case in which overhead is minimized, the optimal processor assignments that we obtain below have properties that suggest that they may also be reasonable approximations to assignments that would minimize other response time functions under reasonable assumptions on communications costs. For example, suppose the response time r for an average query as a function of the number of processors ν used by the query is modeled by

$$r(\nu) = V\nu + \frac{Q}{\nu},$$

where V is the overhead per processor and Q is the processing time for a query on a single processor. In this model we assume that the overhead increases proportionally with the number of processors, and that the processing time is inversely proportional to the number of processors[2]. In the absence of any constraints, r is minimized when the number of processors per query is $\nu^* = \sqrt{Q/V}$. We shall see in (§3) that for our version of the problem, which in some sense minimizes the number of processors used per query (subject to a load balancing constraint), optimal assignments nevertheless have $\approx \sqrt{P}$ processors assigned to each query, where P is the number of processors in the system. If $\sqrt{Q/V} \leq \sqrt{P}$, which is the case if the communication overhead for using all P processors dominates the processing time for a single query, then our optimal solution comes as close as possible (among assignments that balance the workload among processors) to the unconstrained minimum of the alternative objective r.

1.2 Overview

This paper formalizes the problem of assigning the cells of a multidimensional grid to a given number of processors, in order to minimize overhead. In §2 we present a mathematical statement of the problem. In §3, we derive lower bounds on the maximum and on the average number of distinct processors that must appear in the slices while evenly assigning the cells to the processors. §4 and §5 provide optimal assignments that attain the lower bounds in many cases. In §6 and §8, we present other variations of the problem which arise when considering factors such as relative frequencies of access to the different attributes (or dimensions), and relative sizes of cells. §7 presents results of a heuristic assignment procedure applied to some standard benchmarks. Our conclusions and future research directions are contained in §9. In an appendix, we show an integer programming formulation for our basic problem. However the resulting integer programs are quite large, and we were unable to solve any interesting cases using standard software. This computational experience helped to motivate the analytical results of this paper.

2 Basic Mathematical Problem Statement

In this section we state the problem in a mathematical form, using the motivation of §1. First we introduce some notation and state some definitions.

[2]The linear speedup results presented in [7] justify this assertion.

Suppose that we wish to assign the cells of a D-dimensional grid to P processors, and that the dth dimension is partitioned into M_d ranges.

Let $\mathbb{Z}$ denote the integers, and $\mathbb{Z}^D$ the set of D-tuples of integers. If $M \in \mathbb{Z}^D$ is strictly positive, we let $\mathbb{Z}_M^D$ denote the set of D-tuples of integers, with the condition that the dth coordinate is restricted to $\{0, \ldots, M_d - 1\}$. Therefore, each cell of the grid corresponds to a point in $\mathbb{Z}_M^D$. We shall use the shorthand

$$\prod x = \prod_{d=0}^{D-1} x_d \qquad \text{and} \qquad \sum x = \sum_{d=0}^{D-1} x_d$$

for any $x \in \mathbb{Z}^D$ whenever it is convenient.

When considering a single value of the dth attribute, all the data that has the given value for the dth attribute lies in a $D-1$-dimensional "slice" of the grid. This motivates the following definition. The *jth slice of $\mathbb{Z}_M^D$ in dimension d* is defined to be the set

$$S(d, j) := \{x = (x_0, \ldots, x_{D-1}) \in \mathbb{Z}_M^D | x_d = j\}.$$

The slices of $\mathbb{Z}_M^D$ are simply the $D-1$-dimensional subsets obtained by fixing one coordinate. Let $\mathcal{S}$ denote the collection of all slices:

$$\mathcal{S} := \{S(d, j) | d \in \{0, \ldots, D-1\} \text{ and } j \in \{0, \ldots, M_d - 1\}\},$$

and note that $|\mathcal{S}| = \sum M$.

In order to assign the P processors to cells of the grid, we consider an *assignment* to be a function that maps the cells of the grid to the processors, i.e.,

$$a : \mathbb{Z}_M^D \to \mathbb{Z}_P.$$

The problem we will consider is to find an assignment that minimizes an *objective criterion* subject to satisfying a *balancing condition*, both of which will be defined below. We say that a cell $x \in \mathbb{Z}_M^D$ is assigned to p if $a(x) = p$. We also say that a slice $S \in \mathcal{S}$ *contains* p if $a(x) = p$ for some $x \in S$.

Given an assignment a and an arbitrary slice S of $\mathbb{Z}_M^D$, let $\nu(a, S)$ denote the number of *distinct* processors in the slice S; i.e. the number of elements in the image $a(S)$. Given a processor $p \in \mathbb{Z}_P$, let $\lambda(a, p)$ denote the number of cells of $\mathbb{Z}_M^D$ assigned to p, i.e. the number of elements in the inverse image $a^{-1}(p)$.

In the basic problem, the objective criterion is defined using a function θ that specifies either a maximum or an average of the number of processors in the slices. Therefore, we consider $\theta(a)$ such that

$$\theta(a) = \text{ either } \begin{cases} \theta_{\text{ave}}(a) := \operatorname{ave}_{S \in \mathcal{S}} \nu(a, S) \\ \qquad\qquad \text{or} \\ \theta_{\max}(a) := \max_{S \in \mathcal{S}} \nu(a, S). \end{cases} \tag{1}$$

(We let the "ave" operator on a finite set of numbers be the sum of the numbers divided by the cardinality of the set.) Note that if each slice $S \in \mathcal{S}$ has the same frequency of access, then the average number of distinct processors in each slice ($\theta = \theta_{\text{ave}}$) represents the average number of distinct processors used in a typical query. Therefore, if we are interested in minimizing the average query overhead, then we should minimize θ_{ave}. If, on the other hand, we are interested in minimizing the worst case overhead incurred by *any query*, then $\theta_{\max}$ should be minimized.

The first and most basic problem variation that we wish to consider is as follows:

PROBLEM VARIATION 1. *Let the following data be given: a dimension D, a processor count $P \in \mathbb{Z}$, the cardinality of the partitions in each dimension M. (D, P and each of the M_d must be positive.) Find an assignment $a : \mathbb{Z}_M^D \to \mathbb{Z}_P$ that*

$$\textit{minimizes } \theta(a) \qquad (\textit{where } \theta \textit{ is chosen as } \theta_{\text{ave}} \textit{ or } \theta_{\max})$$

subject to the constraint that each processor is assigned an equal number of cells; i.e., $\lambda(a, p)$ is the same for all $p \in \mathbb{Z}_P$.

Any assignment a satisfying the constraint in problem variation 1 is called a *balanced* assignment. Note that a necessary condition for a to be balanced is that $P | \prod M$, in which case the number of balanced assignments that exist is equal to

$$\frac{\left(\prod M\right)!}{\left[\left(\frac{\prod M}{P}\right)!\right]^P}.$$

Complete enumeration is not feasible even for relatively small problems. E.g., if $(D, P, M) = (2, 4, [4, 4])$ then there are $63,063,000$ balanced assignments; while the the number of balanced assignments for the instance $(D, P, M) = (2, 5, [5, 5])$ is $623,360,743,125,120$.

A similar class of data aggregation problems was studied by Helman [12]. Suppose that we replace our notion of "slice" by "arbitrary subset". In symbols, let $\mathcal{U} \subset 2^{\mathbb{Z}_M^D}$ be some arbitrary subcollection of subsets of $\mathbb{Z}_M^D$, and let $\nu(a, U) = |a(U)|$ be the number of points in the image of $U \in \mathcal{U}$ under a. We then let our θ-function be defined as

$$\tilde{\theta}(a) = \text{ either } \begin{cases} \tilde{\theta}_{\text{ave}}(a) & := & \underset{U \in \mathcal{U}}{\text{ave}}\, \nu(a, U) \\ & \text{or} & \\ \tilde{\theta}_{\max}(a) & := & \max\limits_{U \in \mathcal{U}} \nu(a, U). \end{cases}$$

With the θ-function so modified, our problem variation 1 becomes Helman's *K-size aggregation problem (with unit frequencies)*, with $K = \prod M/P$. Helman shows that the K-size aggregation problem (with unit frequencies) is NP-complete.

3 Lower Bounds for Balanced Assignments

In this section we will prove lower bounds on the measures $\theta_{\text{ave}}(a)$ and $\theta_{\max}(a)$ defined in (1) for balanced assignments a (which implies $P|\prod M$). Throughout this section, the following notation is used: σ_p is the number of slices containing processor p and $\sigma_{p,d}$ is the number of slices in dimension d containing processor p. In symbols:

$$\sigma_{p,d} := \left|\left\{ \; S(d,j) \in \mathcal{S} \;\middle|\; \exists\{x \in S(d,j)\} \;\; a(x) = p \;\right\}\right|.$$

$$\sigma_p := \left|\left\{ \; S \in \mathcal{S} \;\middle|\; \exists\{x \in S\} \;\; a(x) = p \;\right\}\right| = \sum_{d=0}^{D-1} \sigma_{p,d}.$$

Recall that $\lambda(a,p)$ is the total number of cells a assigns to the processor p; i.e., $\lambda(a,p) := |a^{-1}(p)|$.

LEMMA 1. *For any processor p and any (not necessarily balanced) assignment a,*

$$\left\lceil D\,[\lambda(a,p)]^{\frac{1}{D}} \right\rceil \leq \sigma_p.$$

Proof: Fix $p \in \mathbb{Z}_P$. Working in each dimension, successively permute the slices $S(d,\cdot)$ so that

$$S(d,j) \text{ contains } p \text{ iff } j < \sigma_{p,d}.$$

(Notice that this operation does not alter any of the $\sigma_{p,d}$.) Then the box bounded by $(\sigma_{p,0},\ldots,\sigma_{p,D-1})$ contains all $\lambda(a,p)$ of the cells assigned to p. Therefore the "volume" $\prod_{d=0}^{D-1}\sigma_{p,d}$ of this box is at least $\lambda(a,p)$, i.e.,

$$\lambda(a,p) \leq \prod_{d=0}^{D-1} \sigma_{p,d}. \tag{2}$$

Taking Dth roots gives

$$(\lambda(a,p))^{\frac{1}{D}} \leq \left(\prod_{d=0}^{D-1} \sigma_{p,d}\right)^{\frac{1}{D}}.$$

Because the arithmetic mean dominates the geometric mean (see Hardy *et al* [11] or Beckenbach and Bellman [3])

$$(\lambda(a,p))^{\frac{1}{D}} \leq \left(\prod_{d=0}^{D-1} \sigma_{p,d}\right)^{\frac{1}{D}} \leq \frac{1}{D}\sum_{d=0}^{D-1} \sigma_{p,d} = \frac{\sigma_p}{D},$$

whence

$$D\,(\lambda(a,p))^{\frac{1}{D}} \leq \sigma_p.$$

Since the right-hand-side of the last inequality is integral, we may take the ceiling of the left-hand-side, finishing the proof. ∎

Letting $\mathbb{Q}$ denote the field of rational numbers, we state a short number theoretic lemma.

LEMMA 2. *If $z \in \mathbb{Z}$ and $r \in \mathbb{Q}_+$, then z^r is either an integer or is irrational.*

Proof: See theorem 16.1.2 of Keng [13]. ∎

3.1 Lower Bounds on θ_{ave}

We shall use the notation

$$\text{arithmetic.mean}(M) = \frac{\sum M}{D}$$

and

$$\text{geometric.mean}(M) = \left(\prod M\right)^{\frac{1}{D}}$$

in the rest of the paper. The following two theorems give lower bounds on the θ-measures for balanced assignments.

THEOREM 3. *Let $\theta_{ave}(\cdot)$ be as defined in (1) and let the assignment a be balanced. Letting*

$$\ell_{ave} := \frac{geometric.mean(M)}{arithmetic.mean(M)} P^{(\frac{D-1}{D})} \qquad and \qquad \ell'_{ave} := \frac{P\left\lfloor D\left(\frac{\prod M}{P}\right)^{\frac{1}{D}}\right\rfloor}{\sum M},$$

we have

$$\ell_{ave} \leq \ell'_{ave} \leq \theta_{ave}(a).$$

Moreover, $\ell_{ave} = \ell'_{ave}$ iff $(\prod M/P)^{\frac{1}{D}}$ is an integer.

Proof: Given an assignment a, let χ_S^p be the characteristic function defined as

$$\chi_S^p = \begin{cases} 1 & \text{if } p \in a(S) \\ 0 & \text{if } p \notin a(S). \end{cases} \tag{3}$$

Recalling that $\nu(a, S)$ is the number of distinct processors in the slice S, we see that

$$\nu(a, S) = \sum_{p=0}^{P-1} \chi_S^p.$$

Also, since σ_p is the total number of slices containing p, we have

$$\sigma_p = \sum_{S \in \mathcal{S}} \chi_S^p,$$

so that

$$\sum_{p=0}^{P-1} \sigma_p = \sum_{p=0}^{P-1} \sum_{S \in \mathcal{S}} \chi_S^p = \sum_{S \in \mathcal{S}} \nu(a, S).$$

Since a is balanced, we know that $\lambda(a, p) = (\prod M)/P$ for each p. Recalling that there are $\sum M$ slices in total, the average number of distinct processors in a slice is expressed as

$$\theta_{\text{ave}}(a) = \frac{\sum_{S \in \mathcal{S}} \nu(a, S)}{\sum M} = \frac{\sum_{p=0}^{P-1} \sigma_p}{\sum M}$$

$$\geq \frac{\sum_{p=0}^{P-1} \left\lceil D\left(\frac{\prod M}{P}\right)^{\frac{1}{D}} \right\rceil}{\sum M} \qquad \begin{pmatrix}\text{by lemma 1, assigning}\\ \lambda(a,p) = (\prod M)/P\\ \text{cells to each processor}\end{pmatrix}$$

$$= \underbrace{\frac{P\left\lceil D\left(\frac{\prod M}{P}\right)^{\frac{1}{D}} \right\rceil}{\sum M}}_{\ell'_{\text{ave}}} \geq \underbrace{\frac{\left(\prod M\right)^{\frac{1}{D}}}{\left(\sum M\right)/D} P^{\left(\frac{D-1}{D}\right)}}_{\ell_{\text{ave}}}.$$

Next notice that $\ell_{\text{ave}} = \ell'_{\text{ave}}$ iff the quantity inside the "ceiling" brackets is an integer. By lemma 2, this occurs iff $(\prod M/P)^{\frac{1}{D}}$ is an integer. As a final remark, notice that $0 \leq \ell'_{\text{ave}} - \ell_{\text{ave}} < P/(\sum M)$. ∎

As §4 and §5 will demonstrate, there are many cases where balanced a exists so that $\ell'_{\text{ave}} = \theta_{\text{ave}}(a)$. However, as proposition 10 will demonstrate, there are cases where $\ell'_{\text{ave}} = \theta_{\text{ave}}(a)$ is not possible for balanced a.

3.2 Lower Bounds on $\theta_{\max}$

At this point we prove a lemma that is used in deriving lower bounds on $\theta_{\max}$. Let

$$\nu_{d,j} = \nu\left(a, S(d,j)\right) \qquad \text{and} \qquad \tilde{\nu}_d = \operatorname*{ave}_j \nu_{d,j} = \frac{\sum_j \nu_{d,j}}{M_d}. \tag{4}$$

LEMMA 4. *For balanced assignments a, $\prod \tilde{\nu} \geq P^{D-1}$.*

Proof: Let χ_S^p be defined as in (3). For fixed d,

$$\sum_{j=0}^{M_d-1} \nu_{d,j} = \sum_{j=0}^{M_d-1} \sum_{p=0}^{P-1} \chi^p_{S(d,j)} = \sum_{p=0}^{P-1} \sigma_{p,d}. \tag{5}$$

Since a is balanced, equation (2) shows that

$$\prod_{d=0}^{D-1} \sigma_{p,d} \geq \lambda(a,p) = \frac{\prod M}{P} \qquad \forall p. \tag{6}$$

Since the set of D-vectors $(\sigma_{p,0}, \ldots, \sigma_{p,D-1})$ that satisfy (6) is convex (see lemma 20 in appendix Appendix C), and the average of a set of vectors is a convex combination of those vectors, we have

$$\prod_{d=0}^{D-1} \frac{\sum_{p=0}^{P-1} \sigma_{p,d}}{P} \geq \frac{\prod M}{P}.$$

From this and (5) we have

$$P^{D-1} \leq \frac{\prod_{d=0}^{D-1}\left(\sum_{p=0}^{P-1}\sigma_{p,d}\right)}{\prod M} = \frac{\prod_{d=0}^{D-1}\left(\sum_{j=0}^{M_d-1}\nu_{d,j}\right)}{\prod M} = \prod_{d=0}^{D-1}\tilde{\nu}_d.$$

∎

Now we derive a lower bound on $\theta_{\max}$.

THEOREM 5. *Let $\theta_{max}(\cdot)$ be as defined in (1) and let the assignment a be balanced. Letting*

$$\ell_{max} := P^{(\frac{D-1}{D})} \qquad \text{and} \qquad \ell'_{max} := \left\lceil P^{(\frac{D-1}{D})} \right\rceil,$$

we have

$$\ell_{max} \leq \ell'_{max} \leq \theta_{max}(a).$$

Proof: By lemma 4, $\prod \tilde{\nu} \geq P^{D-1}$, so there must be at least one dimension d such that $\tilde{\nu}_d \geq P^{(\frac{D-1}{D})}$. Since $\tilde{\nu}_d$ is an average of the number of distinct processors per slice over all the slices in dimension d, one of these slices must have at least $P^{(\frac{D-1}{D})}$ processors in it. Since the number of processors in a slice is integral we have

$$P^{(\frac{D-1}{D})} \leq \left\lceil P^{(\frac{D-1}{D})} \right\rceil \leq \theta_{\max}(a).$$

∎

In sections 4 and 5 we shall see many cases where balanced a exist so that $\ell'_{\max} = \theta_{\max}(a)$. As a negative result, however, note that there are cases where the refined lower bound $\ell'_{\max}$ is not attainable. Clearly, $\ell'_{\text{ave}} \leq \theta_{\text{ave}}(a) \leq \theta_{\max}(a)$ for all balanced a. If $M = (4, 5)$ and $P = 4$ for example, then $\ell_{\max} = \ell'_{\max} = 2$, but $\ell'_{\text{ave}} = 20/9 \approx 2.2$, which implies that 3 is a lower bound on $\theta_{\max}(a)$ for balanced a. (Considering only two dimensional cases where balanced assignments exist, there are four such examples for $P, M_d \leq 5$ and 436 such examples for $P, M_d \leq 32$.)

4 Blocking

This section introduces the concept of *blocking* to construct an assignment a_f based on a special factorization f of P. The blocking approach involves dividing the grid into P blocks and assigning to each block its own processor. Blocking is only possible when the grid can be divided into P identically shaped hyper-rectangular blocks. Furthermore, in order for blocking to yield an optimal solution, P, D and M must be related in very special ways. In particular, P must be representable as the product of D factors, and except for unusual cases, all of these factors must be greater than 1. Therefore, in most instances we require $2^D \leq P$. To appeal to the reader's intuition, we will first describe blocking in two dimensions (see figure 2) and then generalize to arbitrary dimensions.

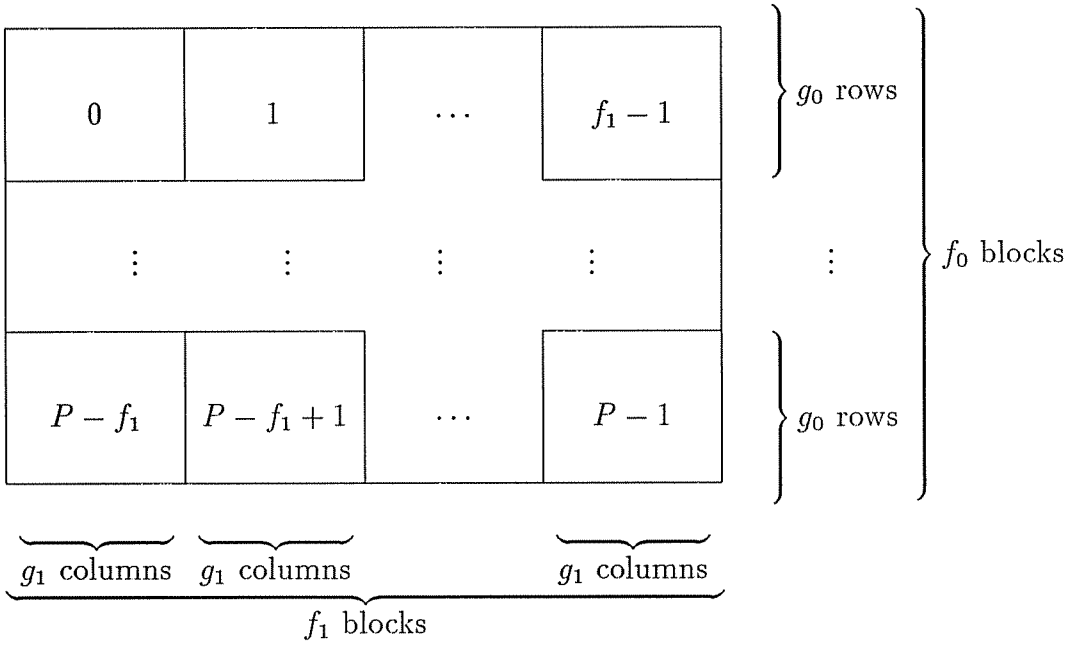

Figure 2: Blocking in two dimensions (processor assignments shown in blocks)

4.1 Two Dimensions

Suppose $P = f_0 f_1$ for positive integer factors f_0 and f_1. Further assume that $f_0 g_0 = M_0$ (the number of rows) and $f_1 g_1 = M_1$ (the number of columns) for integers g_0 and g_1. (This assumes that $f_d | M_d$ for each dimension d.) Consider the grid as a collection of contiguous rectangular blocks, each block having size $g_0 \times g_1$. There are $f_0 f_1 = P$ such blocks. Based on this factorization of P, we construct the assignment a_f by assigning each block to a different processor. Figure 1 is an example of blocking in two dimensions with $f_0 = f_1 = 3$ and $g_0 = g_1 = 2$. Since each block is $g_0 \times g_1$, each processor p is assigned $\lambda(a_f, p) = g_0 g_1$ cells. If the slice S is obtained by fixing a row index to i (i.e. $S = S(0, i)$), then the number of processors in S is given by f_1 (i.e. $\nu(a_f, S) = f_1$). If, on the other hand, the slice S is obtained by fixing a column index to j (i.e. $S = S(1, j)$), then the number of processors in S is given by f_0 (i.e. $\nu(a_f, S) = f_0$). This yields the θ-measures

$$\theta_{\text{ave}}(a_f) = \operatorname*{ave}_{S \in \mathcal{S}} \nu(a_f, S) = \frac{f_1 M_0 + f_0 M_1}{M_0 + M_1}$$

and

$$\theta_{\max}(a_f) = \max_{S \in \mathcal{S}} \nu(a_f, S) = \max\{f_1, f_0\}.$$

4.2 D Dimensions

Now consider the general D-dimensional case. Assume that P is factored as

$$P = \prod_{d=0}^{D-1} f_d \tag{7}$$

and that

$$M_d = f_d g_d \text{ for } d \in \{0, \ldots, D-1\} \tag{8}$$

where

$$f_d \text{ and } g_d \text{ are positive integers for } d \in \{0, \ldots, D-1\}. \tag{9}$$

(Note again that this implies $f_d | M_d$ for all d.) Consider $\mathbb{Z}_M^D$ as a collection of contiguous hyper-rectangular blocks, each block having size $g_0 \times \cdots \times g_{D-1}$. There are $\prod f = P$ such blocks. Based on the factorization f of P given in (7), we construct the assignment a_f by assigning each block to a different processor. Since each block is $g_0 \times \cdots \times g_{D-1}$, each processor p is assigned to $\lambda(a_f, p) = \prod g$ cells. If the slice S is obtained by fixing an index in dimension d to i (i.e. $S = S(d,i)$), then the number of processors in S is given by $\prod_{k \neq d} f_k$ (i.e. $\nu(a_f, S) = \prod_{k \neq d} f_k$). This yields the θ-measures

$$\theta_{\text{ave}}(a_f) = \operatorname*{ave}_{S \in \mathcal{S}} \nu(a_f, S) = \frac{\sum_{d=0}^{D-1} \left(\prod_{k \neq d} f_k \right) M_d}{\sum M}$$

and

$$\theta_{\max}(a_f) = \max_{S \in \mathcal{S}} \nu(a_f, S) = \max \left\{ \prod_{k \neq d} f_k | d = 0, \ldots, D-1 \right\}.$$

4.3 Relative Distances from Lower Bounds for Blocking

We now compare these θ values to the theoretical lower bounds presented in §3. Define the *relative distance from ℓ_{ave}* for any assignment a to be

$$r_{\text{ave}}(a) = \frac{\theta_{\text{ave}}(a) - \ell_{\text{ave}}}{\ell_{\text{ave}}}.$$

Clearly, r_{ave} is a nonnegative function of balanced a with $r_{\text{ave}}(a) = 0$ only if a is optimal for problem variation 1 with $\theta = \theta_{\text{ave}}$.

PROPOSITION 6. *Consider the blocking procedure applied to an instance of problem variation 1 satisfying (7), (8) and (9) and using $\theta = \theta_{ave}$. The assignment a_f produced by the above blocking procedure has relative distance from ℓ_{ave}*

$$r_{\text{ave}}(a_f) = \frac{arithmetic.mean(g) - geometric.mean(g)}{geometric.mean(g)}.$$

In particular, then, $r_{\text{ave}}(a_f) = 0$ iff all the g_d are the same (i.e., each block is a hypercube).

Proof: From above, we have

$$\begin{aligned}\theta_{\text{ave}}(a_f) &= \frac{\sum_{d=0}^{D-1}\left(\prod_{k\neq d} f_k\right) M_d}{\sum M} = \frac{\sum_{d=0}^{D-1}\left(\prod_{k=0}^{D-1} f_k\right) g_d}{\sum M} \\ &= \frac{\left(\prod f\right)\left(\sum g\right)}{\sum M} = \frac{P\left(\sum g\right)}{\sum M}.\end{aligned}$$

From theorem 3, a lower bound on $\theta_{\text{ave}}(a_f)$ is

$$\ell_{\text{ave}} = \frac{D\left(\prod M\right)^{\frac{1}{D}}}{\sum M} P^{(\frac{D-1}{D})} = \frac{PD\left(\frac{\prod M}{P}\right)^{\frac{1}{D}}}{\sum M} = \frac{PD\left(\prod g\right)^{\frac{1}{D}}}{\sum M}.$$

Therefore, after canceling the $P/(\sum M)$ terms, we have

$$r_{\text{ave}}(a_f) = \frac{(\sum g) - D(\prod g)^{\frac{1}{D}}}{D(\prod g)^{\frac{1}{D}}} = \frac{(\sum g)/D - (\prod g)^{\frac{1}{D}}}{(\prod g)^{\frac{1}{D}}} \tag{10}$$

as desired. Clearly, $r_{\text{ave}}(a_f) = 0$ iff arithmetic.mean(g) = geometric.mean(g). This happens iff all of the g_d are in fact equal to the same quantity (see Hardy *et al* [11] or Beckenbach and Bellman [3]). ∎

Similarly we define the relative distance from the tighter lower bound r'_{ave} and state the following:

PROPOSITION 7. *Consider the same problem instance (D, P, M) and assignment a_f as in proposition 6. Then,*

$$r'_{\text{ave}}(a_f) := \frac{\theta_{\text{ave}}(a_f) - \ell'_{\text{ave}}}{\ell'_{\text{ave}}} = \frac{\sum g - \left\lceil D\left(\prod g\right)^{\frac{1}{D}}\right\rceil}{\left\lceil D\left(\prod g\right)^{\frac{1}{D}}\right\rceil}, \tag{11}$$

satisfies

$$0 \underbrace{\leq}_{\text{A}} r'_{\text{ave}}(a_f) \underbrace{\leq}_{\text{B}} r_{\text{ave}}(a_f).$$

Moreover, equality in A *implies that a_f is optimal, and equality in* B *occurs iff $(\prod g)^{\frac{1}{D}}$ is integral.*

Proof: Theorem 3 shows that r'_{ave} is a nonnegative function of balanced a, and that equality in A implies that a_f is optimal. r'_{ave} is dominated by r_{ave} because ℓ'_{ave} is dominated by ℓ_{ave}. The characterization of equality in B follows from lemma 2 by considering equations (10) and (11). ∎

As an example of the utility of proposition 7, consider the case in which the blocks are "nearly hypercubes" in the sense that

$$g_i = \begin{cases} \alpha + 1 & \text{if } i < K \\ \alpha & \text{if } i \geq K \end{cases}$$

for positive integers α and $K < D$. A fairly involved analysis outlined in appendix Appendix D shows that such blockings are always optimal if $D \leq 11$. This takes care of most cases occurring in practice.

Let us now consider the $\theta_{\max}$ case. We define the relative distance from $\ell_{\max}$ for any assignment a to be

$$r_{\max}(a) = \frac{\theta_{\max}(a) - \ell_{\max}}{\ell_{\max}}.$$

$r_{\max}$ is a nonnegative function of balanced a with $r_{\max}(a) = 0$ only if a is optimal for problem variation 1 with $\theta = \theta_{\max}$.

PROPOSITION 8. *Consider the blocking procedure applied to an instance of problem variation 1 satisfying (7), (8) and (9) and using $\theta = \theta_{\max}$. The assignment a_f produced by the above blocking procedure has relative distance from $\ell_{\max}$*

$$r_{\max}(a_f) = \frac{1 \Big/ \left(\min_d f_d\right) - \textit{geometric.mean}(1/f)}{\textit{geometric.mean}(1/f)},$$

where the geometric mean of $1/f$ is to be understood as the geometric mean of the quantities $\{1/f_d | d = 0, \ldots, D-1\}$. In particular, $r_{\max}(a_f) = 0$ iff all of the f_d are the same, i.e., $P = f_0^D$ and $f_0 | M_d$ for all d. (In geometric terms, this means that the grid is subdivided the same number of times in each dimension. This will produce P blocks with the same proportions as the original grid.)

Proof: From above, we have

$$\theta_{\max}(a_f) = \max\left\{\prod_{k \neq d} f_k | d = 0, \ldots, D-1\right\} = \prod f \left(\frac{1}{\min_d f_d}\right).$$

From theorem 5, a lower bound on $\theta_{\max}(a_f)$ is

$$\ell_{\max} = P^{\left(\frac{D-1}{D}\right)} = P\left(\frac{1}{P}\right)^{\frac{1}{D}} = \prod f \left(\frac{1}{\prod f}\right)^{\frac{1}{D}}.$$

Therefore, canceling the $\prod f$ term gives the formula for $r_{\max}$. Clearly, $r_{\max}(a_f) = 0$ iff the geometric mean of the quantities $1/f_d$ is equal to $1/(\min f_d) = \max\{1/f_d\}$. This occurs iff all of the $1/f_d$ (and hence all of the f_d) are the same (see Hardy *et al* [11] or Beckenbach and Bellman [3]). ∎

Similarly we define a tighter relative distance $r'_{\max}$ and state the following:

PROPOSITION 9. *Consider the same problem instance (D, P, M) and assignment a_f as in proposition 8. Then,*

$$r'_{max}(a_f) := \frac{\theta_{max}(a_f) - \ell'_{max}}{\ell'_{max}} = \frac{\prod f \Big/ \left(\min_d f_d\right) - \left\lceil \left(\prod f\right)^{\frac{D-1}{D}} \right\rceil}{\left\lceil \left(\prod f\right)^{\frac{D-1}{D}} \right\rceil},$$

satisfies

$$0 \underbrace{\leq}_{\text{A}} r'_{max}(a_f) \underbrace{\leq}_{\text{B}} r_{max}(a_f).$$

Moreover, equality in A *implies that a_f is optimal, and equality in* B *occurs iff $(\prod f)^{\frac{D-1}{D}}$ is integral.*

Proof: Similar to the proof of proposition 7. ∎

The results of proposition 9 can be used to show that "slightly irregular" subdivisions provide optimal solutions in certain instances. In the two-dimensional case, for example, if $f_0 = f_1 + 1$, then it is clear that the numerator of $r'_{\max}(a_f)$ vanishes, implying the optimality of the corresponding blocking. A more involved analysis shows, however, that this result does not extend to three dimensions when $f_0 - 1 = f_1 = f_2$ is sufficiently large.

4.4 Optimal Blocking for Elongated Grids

We say that a grid is *elongated* if the size of one dimension dominates the sizes of the others by at least P, i.e., for some d, $M_d \geq PM_i$ whenever $i \neq d$. This section develops a lower bound on $\theta_{\text{ave}}(a)$ (for balanced assignments a) which is sometimes tighter than ℓ'_{ave} and is achievable by blocking in elongated grids.

Recall the notation defined in equation (4). Writing θ_{ave} as a function of $\tilde{\nu}$:

$$\theta_{\text{ave}}(a) = \theta_{\text{ave}}(\tilde{\nu}) = \frac{\sum_{d=0}^{D-1} \sum_{j=0}^{M_d-1} \nu_{d,j}}{\sum M} = \frac{\sum_{d=0}^{D-1} M_d \tilde{\nu}_d}{\sum M}.$$

Clearly, the $\tilde{\nu}_d$ are within the bounds $[1, P]$ because they are the average of integers within that same interval. Since the inequality of lemma 4 also must be satisfied, we may deduce that the optimal value of the nonlinear program

$$\text{minimize } \theta_{\text{ave}}(\tilde{\nu}) \quad \text{subject to } \mathbf{1} \leq \tilde{\nu} \leq P\mathbf{1} \quad \text{and} \quad \prod \tilde{\nu} \geq P^{D-1} \tag{12}$$

(with $\mathbf{1} = (1, \ldots, 1)^{\mathsf{T}}$) is a lower bound on the optimal value of problem variation 1. In appendix Appendix C we show that if any there is one dimension i such that $M_i \geq PM_d$ for all $d \neq i$, then the optimal solution of the nonlinear program has $\tilde{\nu}_i = 1$, and $\tilde{\nu}_d = P$ for $d \neq i$. In this case, we see that the only way this can occur is if (for all j) $\nu_{i,j} = 1$ and $\nu_{d,j} = P$ for $d \neq i$. If $P|M_i$, then constructing an assignment that realizes this is trivial. (Simply divide $\mathbb{Z}_M^D$ into P blocks in dimension i and give each block to a different processor.) Letting a_{E} denote this assignment, we summarize with the following:

PROPOSITION 10. *A lower bound for $\theta_{\mathrm{ave}}(a)$ where a is balanced is given by any feasible solution to the nonlinear program (12). Moreover, if, for some i, $M_i \geq PM_d$ for all $d \neq i$, and $P|M_i$, then a_{E} is well-defined and is an optimal assignment.*

Proof: Given above using the results of appendix Appendix C. ∎

To illustrate that this result may differ significantly from the results of the preceding section, we show a class of problems for which a_{E} is well-defined and optimal, but $r'_{\mathrm{ave}}(a_{\mathrm{E}}) \to +\infty$. Let the problem instance (P, D, M) depend on the positive integer n in the following way:

$$\frac{M_0}{n^D P} = M_1 = M_2 = \cdots = M_{D-1}.$$

(For D, P and M_1 all fixed, this defines a sequence of problems indexed by n.) Then proposition 10 shows that a_{E} is well-defined and optimal. However, since

$$\theta_{\mathrm{ave}}(a_{\mathrm{E}}) = \frac{M_0 + (D-1)PM_1}{\sum M} = \frac{(D - 1 + n^D)PM_1}{\sum M}$$

and

$$\ell'_{\mathrm{ave}} = \frac{P\left\lceil D\left(\frac{\prod M}{P}\right)^{\frac{1}{D}}\right\rceil}{\sum M} = \frac{P\left\lceil D\left(n^D M_1^D\right)^{\frac{1}{D}}\right\rceil}{\sum M} = \frac{nDPM_1}{\sum M},$$

we see that

$$r'_{\mathrm{ave}}(a_{\mathrm{E}}) = \frac{\theta_{\mathrm{ave}}(a_{\mathrm{E}}) - \ell'_{\mathrm{ave}}}{\ell'_{\mathrm{ave}}} = \frac{D - 1 + n^D - n}{Dn} \xrightarrow[n\to\infty]{} +\infty,$$

even though a_{E} is optimal.

4.5 Summary of Blocking

Blocking yields optimal solutions for some instances of problem variation 1, albeit in special cases. Of course, from an applications perspective one could attempt to set up the grid so that it matches one of these cases, provided that P, the number of processors, may be written as a product of D factors. The optimal assignments discussed in the next section do not require any factorization properties of P.

5 Diagonal-Based Assignments in Two Dimensions

This section exhibits another class of problems with optimal closed-form solutions. Unlike the blocking assignments of §4, we construct optimal assignments by successively assigning cells on diagonals. The techniques of this section apply only to two dimensional grids.

5.1 The Case $M = (P, P)$

In this subsection we assume that $D = 2$ and $M = (P, P)$, and we give a technique for constructing an optimal "diagonal-based" assignment. Since $D = 2$, we shall call the slices in one dimension "rows" and the slices in the other dimension "columns." Given an integer $\rho \in \{1, \ldots, P\}$, the technique produces a solution a_ρ so that there are exactly ρ processors in each row and $\lceil P/\rho \rceil$ processors in each column. We then show that a_ρ is optimal for this problem if $\rho = \lceil \sqrt{P} \rceil$; i.e., $\theta_{\text{ave}}(a_\rho) = \ell'_{\text{ave}}$ and $\theta_{\max}(a_\rho) = \ell'_{\max}$ (c.f. §3) for $\rho = \lceil \sqrt{P} \rceil$.

5.1.1 A Diagonal Solution Technique

The jth diagonal of $\mathbb{Z}^2_{(P,P)}$ is defined to be the set of cells $x = (i, k)$ that satisfy

$$k - i \equiv j \bmod P. \tag{13}$$

Note that there are exactly P cells in each diagonal. An algorithm for computing an assignment a_ρ is given in figure 3. This algorithm is given inputs P and $\rho \in \{1, \ldots, P\}$ and assigns one diagonal at a time, using the procedure assign.single.diag. The procedure assign.single.diag is given an "initial processor" $p \in \mathbb{Z}_P$, and a column coordinate $j \in \mathbb{Z}_P$, and then assigns each cell in the jth diagonal to a different processor. Note that the algorithm assign.diags takes $O(P^2)$ operations, which is optimal since $\mathbb{Z}^2_{(P,P)}$ has P^2 cells. Figure 4 shows the portion of the assignment created by assign.diags(7, 3) after the first two calls to procedure assign.single.diag. Figure 5 shows the complete assignment.

We now prove that algorithm assign.diags is correct.

PROPOSITION 11 (FULL ASSIGNMENT). *Algorithm assign.diags assigns each cell of $\mathbb{Z}^2_{(P,P)}$ exactly once.*

Proof: When assign.single.diag$(\cdot, j)$ is called, it assigns exactly those cells $x = (i, k) \in \mathbb{Z}^2_{(P,P)}$ that satisfy (13). The proof follows because assign.single.diag$(\cdot, j)$ is called for each $j \in \{0, \ldots, P-1\}$. ∎

PROPOSITION 12. *If the assignment a_ρ has been computed using assign.diags, then for each $(i, j) \in \mathbb{Z}^2_{(P,P)}$ we have:*

$$a_\rho(i + h \bmod P, j + h \bmod P) \equiv a_\rho(i, j) + h \bmod P \qquad \text{for all } h.$$

```
Procedure assign.single.diag(p, j)
    q ← p          ( initial processor index is p )
    k ← j          ( initial column index is j )
    For i = 0, ..., P − 1
        a_ρ(i, k) ← q
        k ← (k + 1) mod P
        q ← (q + 1) mod P


Algorithm assign.diags(P, ρ)
    For j ← 0, ..., P − 1
        assign.single.diag(j mod ρ, j)
```

Figure 3: The algorithm for computing a_ρ

0						
	1					
		2				
			3			
				4		
					5	
						6

(a)

0	1					
	1	2				
		2	3			
			3	4		
				4	5	
					5	6
0						6

(b)

Figure 4: (a) After 1 diagonal assigned; (b) After 2 diagonals assigned

0	1	2	0	1	2	0
1	1	2	3	1	2	3
4	2	2	3	4	2	3
4	5	3	3	4	5	3
4	5	6	4	4	5	6
0	5	6	0	5	5	6
0	1	6	0	1	6	6

Figure 5: An optimal assignment of $\mathbb{Z}^2_{(7,7)}$ with 7 processors

In a rough sense, this means that if one moves h positions along the diagonal, then the processor that is assigned that cell also changes by $h \bmod P$. Note that "moving along the diagonal" of $\mathbb{Z}^2_{(P,P)}$ involves wrap-around $\bmod P$.

Proof of proposition 12: Follows directly from the procedure assign.single.diag. ∎

PROPOSITION 13. *A call to assign.single.diag either adds one new distinct processor to every row (column) or does not change the number of distinct processors in any row (column).*

Proof: The case of columns is the same as for rows. Clearly, the number of distinct processors in each row either stays the same or increases by one, so it suffices to show that the number of distinct processors in one row stays the same iff the number of distinct processors in other rows stays the same. Say (i, j) gets assigned to p during a particular call to assign.single.diag. Then

$$\begin{gathered}\text{the number of distinct processors in row } i \text{ stays the same}\\ \text{iff}\\ p \text{ already in row } i\\ \text{iff}\\ p+h \text{ already in row } i+h \text{ for all } h \text{ with } 0 \le i+h < P\\ \text{iff}\\ \text{the number of distinct processors in row } i+h \text{ stays the same for all } h \text{ with } 0 \le i+h < P.\end{gathered}$$

The middle implication follows from proposition 12. ∎

PROPOSITION 14 (CORRECTNESS). *assign.diags(P, ρ) terminates with ρ distinct processors in each row and $\lceil P/\rho \rceil$ distinct processors in each column.*

Proof: (Rows): The algorithm puts ρ distinct processors in the first row. Proposition 13 then shows that there are ρ distinct processors in each row.

(Columns): By proposition 13 it suffices to show that assign.diags(P, ρ) puts $\lceil P/\rho \rceil$ distinct processors in the first column (the column with index 0). For any column index $j \in \{0, \ldots, P-1\}$ we decompose j uniquely as $j = k\rho + h$ where $k \in \{0, \ldots, \lceil P/\rho \rceil - 1\}$ and $h \in \{0, \ldots, \rho - 1\}$. The algorithm above and proposition 12 give

$$h = a_\rho(0, j) \equiv a_\rho(-j \bmod P, 0) + j \bmod P.$$

Therefore

$$a_\rho(-j \bmod P, 0) \equiv h - j \equiv -k\rho \bmod P.$$

The proof follows because there are exactly $\lceil P/\rho \rceil$ such k. ∎

Note that we may compute $a_\rho(i, j)$ for $(i, j) \in \mathbb{Z}^2_{(P,P)}$ in a constant number of steps by using proposition 12 and the value of $a_\rho(0, \cdot)$:

$$\begin{aligned} a_\rho(i, j) &\equiv \{a_\rho(0, (j-i) \bmod P) + i\} \bmod P \\ &\equiv \{[(j-i) \bmod P] \bmod \rho + i\} \bmod P. \end{aligned}$$

5.1.2 Optimality of the Solution

In this subsection, we look at the solutions produced by the diagonal assignment technique of §5.1.1. We first state a lemma (proved in appendix Appendix B).

LEMMA 15. *Suppose P is a positive integer. Then*

$$round\left(\sqrt{P}\right) = \left\lceil \frac{P}{\lceil\sqrt{P}\rceil} \right\rceil,$$

where round(x) is the function that rounds the real number x to the nearest integer, breaking ties in an arbitrary way when the fractional part is $1/2$.

Suppose we choose

$$\rho = \lceil\sqrt{P}\rceil \tag{14}$$

and that we produce an assignment a_ρ that is as advertised; with ρ processors in each row (i.e. $\nu(a_\rho, S(0,i)) = \rho$ for all rows i) and $\lceil P/\rho \rceil$ processors in each column (i.e. $\nu(a_\rho, S(1,j)) = \lceil P/\rho \rceil$ for all columns j). The lemma then says that the number of processors in each column is round$(\sqrt{P})$.

Consider the optimality of $\theta_{\text{ave}}(a_\rho)$ with the choice of ρ as in (14). Then

$$\theta_{\text{ave}}(a_\rho) = \frac{P\rho + P\lceil P/\rho \rceil}{2P} = \frac{\lceil\sqrt{P}\rceil + \left\lceil \frac{P}{\lceil\sqrt{P}\rceil} \right\rceil}{2} = \frac{\lceil\sqrt{P}\rceil + \text{round}\left(\sqrt{P}\right)}{2}.$$

In this problem, all the $M_d = P$ are the same, so that the lower bound ℓ'_{ave} given in theorem 3 is simplified to

$$\ell'_{\text{ave}} = \frac{P\left\lceil 2\sqrt{P}\right\rceil}{2P} = \frac{1}{2}\left\lceil 2\sqrt{P}\right\rceil.$$

We claim that, in this case, $\theta_{\text{ave}}(a_\rho) = \ell'_{\text{ave}}$, which implies that a_ρ is optimal for problem variation 1.

Proof of the claim: We let $r = \text{round}(\sqrt{P})$ and $\sqrt{P} = r + \varepsilon$ so that $\varepsilon \in [-\frac{1}{2}, \frac{1}{2}]$. We have three cases:

$\varepsilon = \pm\frac{1}{2}$: This case cannot happen, for $\sqrt{P} = r \pm \frac{1}{2}$ would imply $P = r^2 \pm r + \frac{1}{4}$. But P and r are both integers.

$\varepsilon \in (-\frac{1}{2}, 0]$: Then $r = \text{round}(\sqrt{P}) = \lceil\sqrt{P}\rceil$ and $\theta_{\text{ave}} = \lceil\sqrt{P}\rceil$. Since r is an integer and $\varepsilon \in (-\frac{1}{2}, 0]$, $\lceil 2\sqrt{P}\rceil = \lceil 2r + 2\varepsilon\rceil = 2r$. Therefore $\ell'_{\text{ave}} = \frac{1}{2}\lceil 2\sqrt{P}\rceil = \frac{1}{2}(2r) = \lceil\sqrt{P}\rceil = \theta_{\text{ave}}$.

$\varepsilon \in (0, \frac{1}{2})$: Then $r = \text{round}(\sqrt{P}) = \lceil\sqrt{P}\rceil - 1$ and $\theta_{\text{ave}} = \lceil\sqrt{P}\rceil - \frac{1}{2}$. Since r is an integer and $\varepsilon \in (0, \frac{1}{2})$, $\lceil 2\sqrt{P}\rceil = \lceil 2r + 2\varepsilon\rceil = 2r + 1$. Therefore $\ell'_{\text{ave}} = \frac{1}{2}\lceil 2\sqrt{P}\rceil = \frac{1}{2}(2r+1) = r + \frac{1}{2} = \lceil\sqrt{P}\rceil - \frac{1}{2} = \theta_{\text{ave}}$. ∎

Next we use the choice of ρ in (14) to consider the optimality of $\theta_{\max}$. Since round$(\sqrt{P}) \le \lceil\sqrt{P}\rceil$, we have

$$\theta_{\max}(a_\rho) = \max_{S\in\mathcal{S}} \nu(a_\rho, S) = \lceil\sqrt{P}\rceil.$$

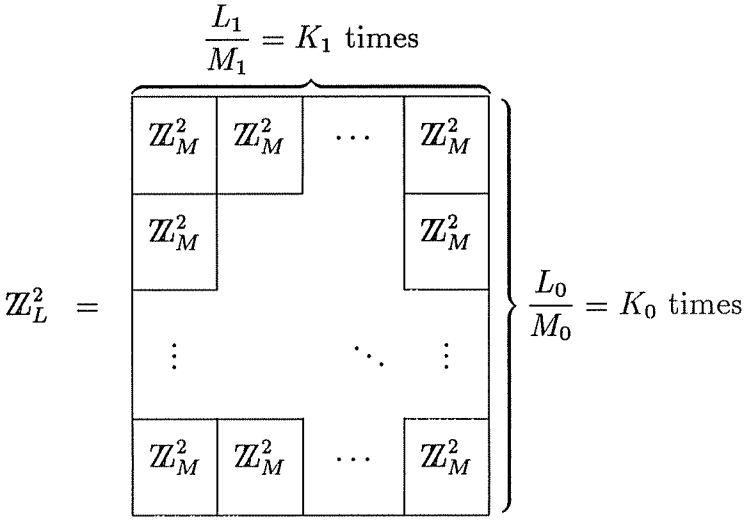

Figure 6: Pictorial representation of tiling in 2 dimensions

This is optimal because theorem 5 shows that

$$\theta_{\max}(a_\rho) = \ell'_{\max}.$$

5.2 Tiling

This subsection introduces the idea of *tiling*. Tiling is the operation of replicating an assignment for $\mathbb{Z}_M^D$ to produce an assignment for $\mathbb{Z}_L^D$ where $M_d | L_d$ for all d. (See figure 6.) Tiling may be done in multiple dimensions, although we will only use the tiling results in two dimensions. Tiling is to be contrasted with blocking (§4) in that each tile contains *all* of the processors, while each block contains a *unique* processor. The results of this subsection are intuitively clear, even though they require a bit of work to prove.

Suppose we have strictly positive $M, L \in \mathbb{Z}^D$ with $L_d = K_d M_d$ for positive integers K_d and for all $d \in \mathbb{Z}_D$. We say that

$$a^L : \mathbb{Z}_L^D \to \mathbb{Z}_P$$

is a tiling with

$$a^M : \mathbb{Z}_M^D \to \mathbb{Z}_P$$

if

$$a^L(x) = a^M(x \bmod M).$$

The first lemma shows that feasibility is maintained when tiling.

LEMMA 16. *Suppose $a^L : \mathbb{Z}_L^D \to \mathbb{Z}_P$ is a tiling with $a^M : \mathbb{Z}_M^D \to \mathbb{Z}_P$. Then a^M is balanced iff a^L is balanced.*

Proof: The integer $\lambda(a,p)$ is defined as the number of things mapped to p by a. By the definition of tiling, we have

$$\lambda(a^L,p) = \left(\prod K\right)\lambda(a^M,p).$$

Therefore, a^L is feasible for problem variation 1 iff $\lambda(a^L,p)$ is the same for all p iff $\lambda(a^M,p)$ is the same for all p iff a^M is feasible for problem variation 1. ∎

The following lemma describes how the function $\theta_{\max}$ is affected by tiling.

LEMMA 17. *If $a^L : \mathbb{Z}_L^D \to \mathbb{Z}_P$ is a tiling with $a^M : \mathbb{Z}_M^D \to \mathbb{Z}_P$, then*

$$\theta_{\max}(a^M) = \theta_{\max}(a^L).$$

Proof: Let $S^L = S(d,j')$ be an arbitrary slice of $\mathbb{Z}_L^D$, and let $j := j' \bmod M_d$. Consider the slice $S^M = S(d,j)$ of $\mathbb{Z}_M^D$. If $p \in a^L(S^L)$, then there is some $x' \in S^L$ with $a^L(x') = p$. If $x := x' \bmod M$, then $x \in S^M$ and $a^M(x) = a^L(x') = p$, so that $p \in a^M(S^M)$; therefore $a^L(S^L) \subset a^M(S^M)$. Because $\nu(\cdot,\cdot)$ is defined as the number of distinct processors in a slice, $\nu(a^L,S^L) \le \nu(a^M,S^M)$, from which we deduce that $\theta_{\max}(a^L) \le \theta_{\max}(a^M)$.

To get the other inequality, let $S^M = S(d,j)$ be an arbitrary slice of $\mathbb{Z}_M^D$, and consider the corresponding slice $S^L = S(d,j)$ of $\mathbb{Z}_L^D$. If $p \in a^M(S^M)$, then there is some $x \in S^M$ with $a^M(x) = p$. But we may also regard x as being in S^L, so $a^L(x) = a^M(x) = p$. Therefore $a^L(S^L) \supset a^M(S^M)$, so that $\nu(a^L,S^L) \ge \nu(a^M,S^M)$, showing $\theta_{\max}(a^L) \ge \theta_{\max}(a^M)$. ∎

5.3 The Case $D = 2$, $M = (bP, cP)$

Suppose $D = 2$ and $M = (bP, cP)$ for positive integers b and c. We will consider each of the two θ-functions of problem variation 1 in turn. We will obtain an exact minimum for $\theta_{\max}$ and an approximate minimum for θ_{ave}.

First suppose that we wish to minimize the $\theta_{\max}(a)$ of problem variation 1 over all balanced assignments a. We will tile $a : \mathbb{Z}^2_{(bP,cP)} \to \mathbb{Z}_P$ with an appropriate $a'_\rho : Z^2_{(P,P)} \to \mathbb{Z}_P$ as computed previously: so we define a_ρ by

$$a_\rho(i,j) = a'_\rho(i \bmod P, j \bmod P).$$

We know from §5.1 how to compute a'_ρ such that $\theta_{\max}(a'_\rho) = \left\lceil\sqrt{P}\right\rceil = \ell'_{\max}$, which is the best possible for both $\mathbb{Z}^2_{(P,P)}$ and $\mathbb{Z}^2_{(bP,cP)}$ by theorem 5. By lemma 17 we know that $\theta_{\max}(a_\rho) = \theta_{\max}(a'_\rho)$, which implies that a_ρ is an optimal assignment for $\mathbb{Z}^2_{(bP,cP)}$.

When $b \gg c$ or $b \ll c$ an assignment a_ρ having $\left\lceil\sqrt{P}\right\rceil$ processors per row and round$(\sqrt{P})$ processors per column, obtained from tiling, does a poor job of minimizing $\theta_{\text{ave}}(a)$. This can be seen by looking at the expression for ℓ_{ave} in theorem 3 and noticing that the ratio of the means is $\ll 1$ in this case. By changing the value of the parameter ρ in the algorithm of §5.1 we can improve $\theta_{\text{ave}}(a_\rho)$. It may be shown that if $\rho = \sqrt{cP/b}$ is an integer that divides P, then $\theta_{\text{ave}}(a_\rho) = \ell_{\text{ave}}$, and so is optimal. Without these assumptions, the analysis becomes much more difficult because of the need for integer variables.

6 A Variation Without Perfect Load Balance

In problem variation 1 we were quite restrictive in our load balancing constraint. This section suggests a relaxation of the load balancing constraint which generalizes problem version 1. This more general problem variation is:

PROBLEM VARIATION 2. *Given $D, P \in \mathbb{Z}$ and $M \in \mathbb{Z}^D$, all of which are positive, and a tolerance $\tau \in \mathbb{Z}_+$. Find $a : \mathbb{Z}_M^D \to \mathbb{Z}_P$ that*

$$\textit{minimizes } \theta(a) \textit{ (from equation (1))}$$

$$\textit{subject to } \max_{p=0}^{P-1} \lambda(a,p) - \min_{p=0}^{P-1} \lambda(a,p) \leq \tau.$$

Notice that if $P | \prod M$ setting $\tau = 0$ gives us problem variation 1. Otherwise $\tau = 1$ is the smallest value of τ for which the problem has a feasible solution. For this reason we define

$$\tau_* := \begin{cases} 0 & \text{if } P | \prod M \\ 1 & \text{otherwise} \end{cases} \tag{15}$$

to represent the smallest τ for which balanced assignments exist. Letting τ be larger allows us to relax the balancing constraint. We shall say that an assignment a is *τ-balanced* if it is feasible for problem variation 2.

In this more general situation, we may seek lower bounds on $\theta_{\text{ave}}(a)$ and $\theta_{\max}(a)$ for τ-balanced a. Note that, since the average number of cells assigned to p is given by $\prod M/P$, any τ-balanced a has

$$\left\lceil \frac{\prod M}{P} - \tau \right\rceil \leq \lambda(a,p) \qquad \forall p. \tag{16}$$

THEOREM 18. *If a is τ-balanced, then*

$$\frac{P \left\lceil D \left\lceil \frac{\prod M}{P} - \tau \right\rceil^{\frac{1}{D}} \right\rceil}{\sum M} \leq \theta_{\text{ave}}(a) \tag{17}$$

and

$$\left\lceil P \left(\frac{\left\lceil \frac{\prod M}{P} - \tau \right\rceil}{\prod M} \right)^{\frac{1}{D}} \right\rceil \leq \theta_{max}(a). \tag{18}$$

Moreover, if a is $\tau_\star$-balanced (c.f. equation (15)) and the nonnegative integers q and $r < P$, are (uniquely) defined to satisfy $\prod M = qP + r$, then

$$\ell'_{ave} := \frac{r\left\lceil D\left\lceil \frac{\prod M}{P}\right\rceil^{\frac{1}{D}}\right\rceil + (P-r)\left\lceil D\left\lfloor \frac{\prod M}{P}\right\rfloor^{\frac{1}{D}}\right\rceil}{\sum M} \leq \theta_{ave}(a). \tag{19}$$

(Note that inequality (19) is tighter than (17) in this instance.)

Proof: The proof of (17) is the same as that of theorem 3, except that the lower bound from (16) is used in the application of lemma 1.

The proof of (18) follows from a straightforward generalization of lemma 4: The inequality (16) is used in (6), yielding the generalized result:

$$\frac{P^D\left\lceil \frac{\prod M}{P} - \tau\right\rceil}{\prod M} \leq \prod \tilde{\nu}.$$

With this generalized lemma, the proof is exactly that already given for theorem 5.

To prove (19) it is first necessary to note that, in the case where $P|\prod M$ (i.e., $\tau_\star = 0$), the ℓ'_{ave} in (19) is equal to the ℓ'_{ave} of theorem 3. If $P \not| \prod M$ (i.e., $\tau_\star = 1$), then we know that exactly r processors are assigned to $\lambda(a,p) = \lceil (\prod M)/P \rceil$ cells, while the other $P - r$ processors are assigned to $\lambda(a,p) = \lfloor (\prod M)/P \rfloor$ cells. The proof now follows that of theorem 3 except that these values for $\lambda(a,p)$ are used in the application of lemma 1. ∎

As might be expected, techniques for generating provably optimal solutions for problem variation 2 (for $\tau > 0$) are more difficult than for problem variation 1. To give the reader the flavor of what is possible, we shall consider a two dimensional problem in which we are seeking to "extend" a blocking.

Consider a two dimensional grid of size $M_0 \times M_1$ where we have $M_0 M_1 = k^2 P$ for some positive integer $k | M_d$ for all d. Using θ_{ave} as our objective, proposition 6 shows that an optimal blocking is easily computed using the blocking technique. Now suppose we wish to enlarge the grid to be $(M_0 + e) \times M_1$, i.e., we augment $e > 0$ rows onto the original grid. Assume that this enlargement adds at most P cells, so that $eM_1 \leq P$. (If P is large relative to M_1, this may allow a significant enlargement.) Note that if $eM_1 < P$, then a balanced assignment is not possible and $\tau_\star = 1$.

We outline a way to extend the original (optimal) blocked assignment into an assignment that is optimal for the enlarged grid: Let a *column block* represent the set of columns in the same block. In the extension, under each column block, there are ke cells; all unassigned. The number of distinct processors assigned in the blocks directly above this is M_0/k. The assumptions $eM_1 \leq P$ and $M_0 M_1 = k^2 P$ show $ke \leq kP/M_1 = M_0/k$. Therefore, it is possible to select ke distinct processors from the above blocks (in the original blocked grid). These ke processors are each assigned to exactly one cell in the augmented rows. This is then done for each of the column blocks.

If a_e denotes the assignment from the previous paragraph, it is possible to show that $\theta_{\text{ave}}(a_e) = \ell'_{\text{ave}}$. We outline this for the case $eM_1 < P$ (it remains true if $eM_1 = P$ as the reader is invited to verify): First, note that the remainder term r in (19), is given by $r = eM_1$. Then, considering the formula for ℓ'_{ave}, show the inequalities $k^2 < (M_0+e)M_1/P = k^2 + eM_1/P < k^2 + 1$ and $2k < 2\sqrt{k^2+1} < 2k+1$. This leads to the conclusion that

$$\ell'_{\text{ave}} = \frac{eM_1(2k+1) + (P - eM_1)2k}{M_0 + e + M_1} = \frac{2\dfrac{M_0M_1}{k} + eM_1}{M_0 + e + M_1} = \theta_{\text{ave}}(a_e),$$

the final equality following because the final e rows of the grid each have M_1 distinct processors assigned in them.

7 Performance of a Heuristic

Ghandeharizadeh [9] describes a heuristic for assigning the cells of a D-dimensional grid to the processors in the system. The heuristic creates an assignment by approximating the blocking procedure of §4 and using θ_{ave} as our measure. The results presented below show that approximate solutions are obtainable and that the lower bound can be used as a tool for evaluating the solution produced by a heuristic method.

If a good blocked assignment is possible, the heuristic creates the blocked assignment. Notice that a blocked assignment implies that each processor is assigned the same number of cells. Recall also that, roughly speaking, a "good" blocked assignment is one in which the blocks are "nearly square" (see propositions 6 and 7). When blocking does not yield a good assignment, the heuristic uses a divide and conquer approach similar to the one used at the end of §6 to assign the cells. It creates a good blocked assignment on a large subgrid, leaving the cells on the edges unassigned. The remaining cells are then assigned while trying to satisfy the conflicting goals of keeping both θ_{ave} and the imbalance τ between processors small. We refer the interested reader to Ghandeharizadeh [9] for the complete description of the heuristic.

In each of the experiments on which we report, MAGIC scanned a 100,000 tuple relation and used two attributes of this relation to construct a two-dimensional grid on the relation. The characteristics of the relation were based on the standard Wisconsin Benchmark relations [4].

In the first experiment, the heuristic assigned the cells of a 32×31 grid. Table 1 presents a number of measurements for the problem instance and the assignment a_{H} as a function of the number of processors P. The rational data in the table is exact, and decimal data is accurate to three significant digits. For this grid the heuristic created $\tau_\star$-balanced assignments. Therefore, we compare $\theta_{\text{ave}}(a_{\text{H}})$ with the lower bound ℓ'_{ave} from (19). (Recall that if $P|\prod M$, i.e., $\tau_\star = 0$, then ℓ'_{ave} is the same as the quantity defined in theorem 3; this occurs here for $P = 8, 16, 32$.) The last two columns show the absolute and relative gaps between the average number of processors used by the heuristic and the lower bound. Note that the relative gap varies between 0 and 15%. These results indicate that the heuristic approximates the theoretical lower bound quite well.

Procs P	Lower Bound ℓ'_{ave}	Heuristic $\theta_{\text{ave}}(a_{\text{H}})$	Absolute gap $\theta_{\text{ave}}(a_{\text{H}}) - \ell'_{\text{ave}}$	Relative gap $\frac{\theta_{\text{ave}}(a_{\text{H}}) - \ell'_{\text{ave}}}{\ell'_{\text{ave}}}$
8	184/63	$198/63 \approx 3.14$	0.222	0.0761
10	200/63	$230/63 \approx 3.65$	0.476	0.150
16	256/63	$268/63 \approx 4.25$	0.190	0.0469
20	292/63	$300/63 \approx 4.76$	0.127	0.0274
32	384/63	$400/63 \approx 6.35$	0.254	0.0417
64	512/63	$536/63 \approx 8.51$	0.381	0.0469
128	768/63	$784/63 \approx 12.4$	0.254	0.0208
256	1024/63	$1024/63 \approx 16.3$	0	0

Table 1: A 32×31 grid

The same experiment was conducted on a 65×16 grid. Table 2 presents the accuracy of the heuristic for this case. As with the previous grid, the heuristic produced $\tau_\star$-balanced assignments, making ℓ'_{ave} a lower bound on $\theta_{\text{ave}}(a_{\text{H}})$. We have $\tau_\star = 0$ for $P = 8, 10, 16, 20$ and $\tau_\star = 1$ otherwise. Again, the heuristic approximates the lower bound with a high accuracy. In over half of the cases an optimal assignment was found.

8 More General Variations of the Problem

In this section we motivate and formalize other variations of our basic problem that may be used to model our database application more accurately.

Thus far, we have assumed that each slice of a grid has the same frequency of access as any other slice. Even if the frequency of access for each partitioning attribute is not the same, equal slice frequency holds because the MAGIC partitioning strategy constructs a grid based on the frequency of access to each partitioning attribute. This results in the number of slices along each attribute (or dimension) being proportional to the frequency of access to that attribute, resulting in the same frequency of access to each slice. To illustrate this, recall the EMP relation. If the Social_Security attribute has an 80% frequency of access and the Salary attribute the remaining 20%, MAGIC declustering constructs a grid with four times as many slices in the Salary dimension as in the Social_Security dimension (see Figure 7). Thus, a grid is created in which each slice has the same frequency of access as any other slice.

However, in some applications, a slice of a grid will have a different frequency of access than some other slice of the same attribute. For example, the employees in the EMP relation who earn less than \$10K might be accessed more frequently than the high level mangers who earn more than \$40K because of more frequent inter-departmental changes, salary changes,

Procs P	Lower Bound ℓ'_{ave}	Heuristic $\theta_{\text{ave}}(a_{\text{H}})$	Absolute gap $\theta_{\text{ave}}(a_{\text{H}}) - \ell'_{\text{ave}}$	Relative gap $\frac{\theta_{\text{ave}}(a_{\text{H}}) - \ell'_{\text{ave}}}{\ell'_{\text{ave}}}$
8	184/81	$200/81 \approx 2.47$	0.198	0.0870
10	210/81	$210/81 \approx 2.59$	0	0
16	272/81	$272/81 \approx 3.36$	0	0
20	300/81	$300/81 \approx 3.70$	0	0
32	384/81	$400/81 \approx 4.94$	0.198	0.0417
64	528/81	$528/81 \approx 6.52$	0	0
128	768/81	$784/81 \approx 9.68$	0.198	0.0208
256	1040/81	$1040/81 \approx 12.8$	0	0

Table 2: A 65×16 grid

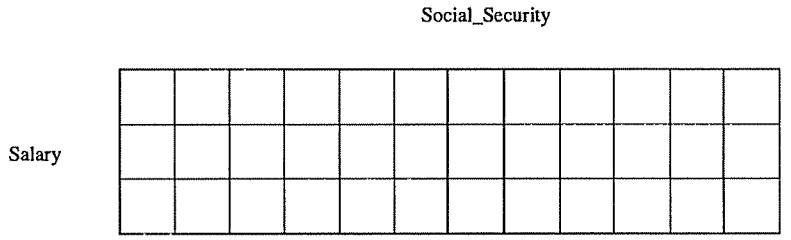

Figure 7: A 2-dimensional grid with 80% / 20% frequencies of access

etc. This property can be incorporated into our mathematical model by assigning frequencies to each slice; i.e., we define a function $\phi : \mathcal{S} \to \mathbb{Q}_+$, mapping the slices to the nonnegative rationals. Then the following modification to the θ-function is made:

$$\theta^\phi(a) = \text{ either } \begin{cases} \theta^\phi_{\text{ave}}(a) & := & \underset{S\in\mathcal{S}}{\text{ave}}\, \phi(S)\nu(a,S) \\ & \text{or} & \\ \theta^\phi_{\max}(a) & := & \underset{S\in\mathcal{S}}{\max}\, \phi(S)\nu(a,S). \end{cases} \tag{20}$$

Furthermore, in our database application, the constraints in problem variation 2 are used to ensure that the number of cells assigned to processor p (which is $\lambda(a,p)$) differs from the number of cells assigned to processor q (which is $\lambda(a,q)$) by no more than some tolerance τ. This is one method to evenly divide the cells of the database among the processors. However, this assumes that each cell of the database contains the same number of tuples—an inaccurate assumption in most cases. This is especially true when MAGIC partitions a relation using two or more correlated attributes. In this case, some cells may contain significantly more tuples than the other cells. For example, if MAGIC partitions the EMP relation using the Salary and Age attributes, which are usually correlated with the older employees earning higher salaries, the cells along a diagonal of the grid will contain more tuples than the other cells. This is incorporated into our model by assigning a weight to each cell; i.e., we are given $\omega : \mathbb{Z}_M^D \to \mathbb{Q}_+$, and require a nearly balanced distribution of the weights among the processors. We recall the old definition of $\lambda(a,p)$ in symbols,

$$\lambda(a,p) = |a^{-1}(p)|,$$

in order to motivate a weighted definition of the load:

$$\lambda^\omega(a,p) = \sum_{x\in a^{-1}(p)} \omega(x).$$

As the original "load" $\lambda(a,p)$ was the number of things that mapped to p, the "weighted load" $\lambda^\omega(a,p)$ is the total "weight" of things that map to p. Therefore, we define the "load imbalance function" γ^ω by

$$\gamma^\omega(a) := \max_{p=0}^{P-1} \lambda^\omega(a,p) - \min_{p=0}^{P-1} \lambda^\omega(a,p). \tag{21}$$

Incorporating these modifications into our problem yields the following more general version of the problem.

PROBLEM VARIATION 3. *Given $D, P \in Z$ and $M \in \mathbb{Z}^D$, all of which are positive, a tolerance $\tau \in \mathbb{Q}_+$, a frequency function $\phi : \mathcal{S} \to \mathbb{Q}_+$, and a weight function $\omega : \mathbb{Z}_M^D \to \mathbb{Q}_+$. Find $a : \mathbb{Z}_M^D \to \mathbb{Z}_P$ that*

minimizes $\theta^\phi(a)$ (from equation (20))

subject to $\gamma^\omega(a) \le \tau$ (from equation (21)).

Note that if $\phi(S) = 1$ for all S and $\omega(x) = 1$ for all x, then problem variation 3 reduces to problem variation 2. In this way, problem variation 3 contains problem variation 2 as a special case. Also note that it is more difficult to choose τ *a priori* in order to make an instance of problem variation 3 feasible for a general weight function ω than it is for the special unit weight function. (Problem variation 2 was feasible with $\tau \geq 0$ or 1, depending on whether or not $P|\prod M$.)

We make the observation that problem variation 3 is NP-hard, even in the case where $\phi(S) = 1$ for each S. This is observed by reducing the *partition problem* to a special case of problem variation 3. The partition problem is known to be NP-complete (see Garey and Johnson [8]), and may be stated as follows: given a finite set X and a positive integer size $s(x)$ for each $x \in X$, does there exist a subset $Y \subset X$ such that $\sum_{x \in Y} s(x) = \sum_{x \notin Y} s(x)$? Given an instance of the partition problem $(X, s(\cdot))$, we construct an instance of problem variation 3 by setting $P = 2$, $D = 1$, $M = |X|$, $\tau = 0$, $\omega(x) = s(x)$ for all x and $\phi(S) = 1$ for all $S \in \mathcal{S}$. The partition question reduces to the question "does there exist a feasible solution to this instance of problem variation 3?"

Our database application involves balancing two conflicting goals: reduce the "overhead" involved with doing a sequence of queries while balancing the "load" of tuples across all of the processors. Up until this time we have considered feasible solutions as those assignments a which (approximately) balance the load, and minimized the overhead of feasible a. (In symbols: minimize $\theta^{\phi}(a)$ subject to $\gamma^{\omega}(a) \leq \tau$.) There are other ways to formulate the problem using the same functions defined above. One problem variation involves switching the roles of the constraint and the objective function.

PROBLEM VARIATION 4. *Given $D, P \in Z$ and $M \in \mathbb{Z}^D$, all of which are positive, a tolerance $\tau \in \mathbb{Q}_+$, a frequency function $\phi : \mathcal{S} \to \mathbb{Q}_+$, and a weight function $\omega : \mathbb{Z}_M^D \to \mathbb{Q}_+$. Find $a : \mathbb{Z}_M^D \to \mathbb{Z}_P$ that*

$$\textit{minimizes } \gamma^{\omega}(a) \textit{ (from equation (21))}$$

$$\textit{subject to } \theta^{\phi}(a) \leq \tau \textit{ (from equation (20)).}$$

Again, it may be difficult to choose τ *a priori* in order to ensure that an instance of problem variation 4 is feasible for a general frequency function ϕ. However, if $\phi(S) = 1$ for all $S \in \mathcal{S}$, then we may take some motivation from §3 (theorems 3 and 5) and choose

$$\tau = \begin{cases} \ell'_{\text{ave}} & \text{if } \theta = \theta_{\text{ave}} \\ \ell'_{\max} & \text{if } \theta = \theta_{\max}. \end{cases} \tag{22}$$

This is well-motivated in the following sense:

> Suppose the weight function, in addition to the frequency function, is trivial, i.e., $\omega(x) = 1$ for all $x \in \mathbb{Z}_M^D$. Suppose (D, P, M) is an instance of problem 1 where

the optimal value $\theta^*_{\text{ave}} = \ell'_{\text{ave}}$ (or $\theta^*_{\max} = \ell'_{\max}$). Then there is an assignment a^* such that

$$\theta(a^*) = \ell' \qquad \text{(in the max or ave case)}$$

and

$$\gamma(a^*) = 0.$$

Using the choice of τ in (22) means that such a a^* is optimal for problem variation 4. Moreover, any optimal assignment for problem variation 4 is an optimal assignment for problem variation 1, because $\gamma(a) \geq 0$.

Another problem variation is stated as a feasibility problem:

PROBLEM VARIATION 5. *Given $D, P \in Z$ and $M \in \mathbb{Z}^D$, all of which are positive, two tolerances $\tau_\gamma, \tau_\theta \in \mathbb{Q}_+$, a frequency function $\phi : \mathcal{S} \to \mathbb{Q}_+$, and a weight function $\omega : \mathbb{Z}^D_M \to \mathbb{Q}_+$. Find $a : \mathbb{Z}^D_M \to \mathbb{Z}_P$ such that the following criteria are satisfied:*

$$\theta^\phi(a) \leq \tau_\theta \qquad \text{and} \qquad \gamma^\omega(a) \leq \tau_\gamma$$

(from equations (20) and (21), respectively).

For general frequency and weight functions ϕ and ω, the previous problems present a difficulty in choosing τ large enough to ensure feasibility, while choosing it small enough to yield a useful result. One way to avoid this problem is to put the effects of θ^ϕ and of γ^ω into one objective function for an unconstrained minimization problem.

PROBLEM VARIATION 6. *Given $D, P \in Z$ and $M \in \mathbb{Z}^D$, all of which are positive, a frequency function $\phi : \mathcal{S} \to \mathbb{Q}_+$, a weight function $\omega : \mathbb{Z}^D_M \to \mathbb{Q}_+$, and an objective function $f : \mathbb{Q}^2 \to \mathbb{R}$. Find $a : \mathbb{Z}^D_M \to \mathbb{Z}_P$ that minimizes*

$$F(a) := f\left(\theta^\phi(a), \gamma^\omega(a)\right)$$

where θ^ϕ and γ^ω are defined above.

Of course, we must choose the objective function f in such a way so that unconstrained minimizers of F provide a good compromise between our conflicting goals. f should also be chosen so that F has properties that facilitate the computational solution of problem variation 6. The choice of such f is another topic which we shall not treat in this paper. Due to the lack of constraints, problem variation 6 also has the benefit of yielding more naturally to probabilistic algorithms like simulated annealing or genetic algorithms.

9 Conclusions

In this paper, we have analyzed a class of combinatorial optimization problems occurring in parallel database design. In order to obtain the best response time and throughput from the parallel database system, the cells must be assigned to the processors such that: 1) the

overhead associated with utilizing parallelism to execute a query is minimized, and 2) the workload of a relation is evenly distributed across the processors in the system.

Analytically, we established lower bounds on the maximum and average number of processors in the slices of a multidimensional grid. These bounds are summarized in table 3. Using these lower bounds, we developed techniques that produce optimal assignments for special problem classes. Table 4 summarizes the major problem classes for which we can construct optimal solutions. The first column defines the problem class by stating the characteristics of the grids in the class. The second column lists the θ-measures which can be optimized for the class. The third column describes the optimal assignment and refers to the section in which it was developed.

The solution techniques we developed can be used in several ways. The study of blocked (§4) and diagonal based (§5) assignments reveals characteristics of grids for which optimal blocked solutions are easily produced. This knowledge may be used to improve the MAGIC partitioning strategy. The grid construction phase of MAGIC declustering could attempt to create grids with these desirable characteristics, leading to easily computable optimal or nearly optimal assignments. These analytic solutions suggest *ad hoc* methods that may apply to more general problems. The heuristic of §7 which uses the ideas of blocked assignments (§4) is an example of this. The lower bounds of §3 were used to measure the solutions produced by the heuristic.

Type of Assignment	Bound
balanced $\left(q = \frac{\prod M}{P}\right)$	$\frac{\text{geometric.mean}(M)}{\text{arithmetic.mean}(M)} P^{\frac{D-1}{D}} \leq \frac{P\left\lceil Dq^{\frac{1}{D}}\right\rceil}{\sum M} \leq \theta_{\text{ave}}$ $\left\lceil P^{(\frac{D-1}{D})}\right\rceil \leq \theta_{\max}$
$\tau_\star$-balanced $\left(q = \frac{\prod M}{P} \text{ and } r = \prod M \bmod P\right)$	$\frac{r\left\lceil D\lceil q\rceil^{\frac{1}{D}}\right\rceil + (P-r)\left\lceil D\lfloor q\rfloor^{\frac{1}{D}}\right\rceil}{\sum M} \leq \theta_{\text{ave}}$
τ-balanced $\left(q = \frac{\prod M}{P}\right)$	$\frac{P\left\lceil D\lceil q-\tau\rceil^{\frac{1}{D}}\right\rceil}{\sum M} \leq \theta_{\text{ave}}$ $\left\lceil P\left(\frac{\lceil q-\tau\rceil}{\prod M}\right)^{\frac{1}{D}}\right\rceil \leq \theta_{\max}$
balanced (elongated grid) $M_0 \geq PM_d; d = 1, 2, \ldots, D-1$	$\frac{M_0 + P\sum_{d=1}^{D-1} M_d}{\sum M} \leq \theta_{\text{ave}}$

Table 3: Summary of lower bounds

Type of Grid	θ-measure	Type of Optimal Solution
$M = (P, P)$	θ_{ave}, $\theta_{\max}$	Diagonal (§5)
$M = (bP, cP)$	$\theta_{\max}$	Tiling with diagonal (§5.3)
$M = P^{\frac{1}{D}}(g_0, \ldots, g_{D-1})$	$\theta_{\max}$	P proportional blocks (proposition 6)
$M = g_0(f_0, \ldots, f_{D-1})$ $\prod f = P$	θ_{ave}	P hypercubical blocks (proposition 8)
$M = (bP, M_1, \ldots, M_{D-1})$ $b \geq M_d; d = 1, 2, \ldots, D-1$ (elongated grid)	θ_{ave}	P blocks of size $b \times M_1 \times \ldots \times M_{D-1}$ (§4.4)

Table 4: Summary of optimal solutions (principal cases)

Appendix A An Integer Programming Formulation

Here we point out that integer linear programming (IP) techniques may be used for problem variation 2 with $\tau = \tau_\star$ (equation (15)). The number of binary variables in our formulation is $P \prod M$. Due to this large number of binary variables, we were unable to solve the problem with a standard IP code. This was a motivating force for the closed form solutions in §4 and §5.

We now briefly sketch the IP formulation. For the assignment $a : \mathbb{Z}_M^D \to \mathbb{Z}_P$ the constraints that force a to be a $\tau_\star$-balanced assignment are constraints in the classical *transportation problem.* In addition to these transportation constraints, we must augment the system with extra constraints and variables in order to be able to model the variables $\nu = (\nu_{d,j})$ (c.f. equation (4)). Once we have represented the variables ν, θ_{ave} is simply a linear function of ν, while $\theta_{\max}$ can be modeled by adding one additional variable ε, adding the constraints $\varepsilon \geq \nu_{d,j}$, and minimizing ε.

The θ_{ave} formulation has $P \prod M + P \sum M$ variables and $\prod M + P + P \sum M$ constraints, not including nonnegativity and integrality constraints on $P \prod M$ of the variables. The $\theta_{\max}$ formulation has $P \prod M + P \sum M + 1$ variables, and $\prod M + P + P \sum M + \sum M$ constraints, not including nonnegativity and integrality constraints on $P \prod M$ of the variables. The large number of variables that must be restricted to integer values makes these formulations difficult for the "off the shelf" techniques we tried. The largest problem of the form $D = 2$, $M = (P, P)$ that we were able to solve using the GAMS (Brooke *et al* [6]) ZOOM module in less than one-half hour was the case where $P = 5$. Clearly, this is not a reasonable alternative for real problems in which the M_i are larger by one or two orders of magnitude.

Appendix B A Technical Lemma on Rounding

In this appendix we prove the technical lemma already stated as lemma 15. For completeness we again state the lemma.

LEMMA 19. *Suppose P is a positive integer. Then*

$$round\left(\sqrt{P}\right) = \left\lceil \frac{P}{\lceil \sqrt{P} \rceil} \right\rceil,$$

where round(x) is the function that rounds the real number x to the nearest integer, breaking ties in an arbitrary way when the fractional part is $1/2$.

Proof: Let the positive integer $r = \text{round}(\sqrt{P})$ and let ε be the fractional portion so that

$$\sqrt{P} = r + \varepsilon \qquad \text{and} \qquad P = (r + \varepsilon)^2,$$

where $-\frac{1}{2} \leq \varepsilon \leq \frac{1}{2}$. If $\varepsilon = \pm\frac{1}{2}$ then $P = r^2 \pm r + \frac{1}{4}$, which is not integral. Therefore, $-\frac{1}{2} < \varepsilon < \frac{1}{2}$. We take two cases separately.

In the first case, suppose the round function does not round down, i.e., $-\frac{1}{2} < \varepsilon \leq 0$. Then $\left\lceil \sqrt{P} \right\rceil = r$ and it suffices to show the following inequality:

$$r - 1 < \frac{(r+\varepsilon)^2}{r} \leq r, \tag{23}$$

because the fraction is equal to $P \Big/ \left\lceil \sqrt{P} \right\rceil$ and $r = \text{round}(\sqrt{P})$. The left inequality of (23) follows by the following reasoning:

$$\begin{gathered} r - 1 < (r+\varepsilon)^2/r \\ \text{iff} \\ r^2 - r < (r+\varepsilon)^2 = r^2 + 2r\varepsilon + \varepsilon^2 \\ \text{iff} \\ 0 < r + 2r\varepsilon + \varepsilon^2 = \underbrace{r}_{\geq 1}\underbrace{(1+2\varepsilon)}_{>0} + \underbrace{\varepsilon^2}_{\geq 0}. \end{gathered}$$

The right inequality of (23) follows because it is equivalent with $(r+\varepsilon)^2 \leq r^2$, which is true because $\varepsilon \leq 0$.

In the other case, the round function rounds down, i.e., $0 < \varepsilon < \frac{1}{2}$. Then $\left\lceil \sqrt{P} \right\rceil = r + 1$ and it suffices to show the following inequality:

$$r - 1 < \frac{(r+\varepsilon)^2}{r+1} \leq r, \tag{24}$$

because the fraction is equal to $P \Big/ \left\lceil \sqrt{P} \right\rceil$ and $r = \text{round}(\sqrt{P})$. The left inequality of (24) follows because it is equivalent with $r^2 - 1 < r^2 + 2r\varepsilon + \varepsilon^2$, which is true because $r \geq 1$ and $\varepsilon > 0$. The right inequality of (24) follows by a more complicated argument:

$$\frac{(r+\varepsilon)^2}{r+1} \leq r \quad \text{iff} \quad r^2 + 2r\varepsilon + \varepsilon^2 \leq r^2 + r \quad \text{iff} \quad \text{LHS} := 2r\varepsilon + \varepsilon^2 \leq r.$$

We finish the proof by showing that LHS $\leq r$ when $P = (r+\varepsilon)^2$ is an integer. Since r is an integer, LHS $= 2r\varepsilon + \varepsilon^2 = (r+\varepsilon)^2 - r^2 = P - r^2$ is also an integer. We know that $2r\varepsilon \leq r$ by our choice of ε, and therefore, LHS $= 2r\varepsilon + \varepsilon^2 \leq r + \frac{1}{4}$.Since both LHS and r are integral, we must have LHS $\leq r$. ∎

We note that this lemma is not true for general $P \in \mathbb{R}_+$, as illustrated by the example $P = (1.49)^2$.

Appendix C Application of Convex Programming

This appendix applies the theory of convexity and convex programming to problems encountered earlier in the paper. First, we prove that that the set of D-vectors that satisfy (6) is convex and that the nonlinear program (12) is a convex program. Then we will address the optimality of (12) in the case of an elongated grid (cf. §4.4).

LEMMA 20. *Let* $C := \{z \in \mathbb{R}_+^D | \prod z \geq K\}$ *for some constant* $K \in \mathbb{R}$. *Then* C *is convex.*

Before giving the proof, we note that $C = \{z \in \mathbb{R}_+^D | \check{h}(z) = \prod z \geq K\}$ where $\check{h}$ is *not concave.*

Proof: If $K \leq 0$, then $C = \mathbb{R}_+^D$, which is convex. If, on the other hand, $K > 0$, then

$$C = \{z \in \mathbb{R}_+^D | h(z) = \sum_{d=0}^{D-1} \ln(z_d) \geq \ln K\}.$$

Since h is the sum of concave functions, h is concave and so C is convex. ∎

We now consider the optimality conditions of the convex program (12) in the case where one dimension is much larger than the others. Without loss of generality, we assume that $M_0 \geq PM_d$ for all $d > 0$. Recall the convex program (12):

$$\underset{\tilde{\nu} \in \mathbb{R}^D}{\text{minimize}} \quad \frac{\sum_{d=0}^{D-1} M_d \tilde{\nu}_d}{\sum M} \quad \text{subject to} \quad \mathbf{1} \leq \tilde{\nu} \leq P\mathbf{1} \quad \text{and} \quad \prod \tilde{\nu} \geq P^{D-1}$$

where $\mathbf{1} = (1, \ldots, 1)^T$. The objective function is linear in $\tilde{\nu}$ and the feasible region is convex by lemma 20. We claim that the feasible point $\bar{\nu}$ given by

$$\bar{\nu}_0 = 1 \qquad \text{and} \qquad \bar{\nu}_d = P \quad \text{for } d > 0$$

satisfies the optimality conditions, implying that $\bar{\nu}$ is an optimal solution.

Proof of claim: The gradient of the nonlinear constraint at $\bar{\nu}$ is

$$(P^{D-1}, P^{D-2}, P^{D-2}, \ldots, P^{D-2}).$$

Choosing the multiplier $M_0 / \left(P^{D-1} \sum M\right)$ for the nonlinear constraint, we find that the reduced gradient is non-positive and has first component 0. ∎

It is interesting to note that many of the results of §3 may be proved by considering the optimality of similar convex programs.

Appendix D "Almost Hypercubical" Blocking

This appendix shows that "almost hypercubical" blocking yields optimal blockings in "small" dimensions, or in any dimension if the grid is large enough. Recall that blocking is applicable whenever we can factor P as (7) and maintain (8) and (9). We use proposition 7 and consider the case in which blocking is applied with

$$g_i = \begin{cases} \alpha + 1 & \text{if } i < K \\ \alpha & \text{if } i \geq K \end{cases} \tag{25}$$

for positive integers α and $K < D$. Blockings that satisfy (25) are called *almost hypercubical.*

We first state a proposition that characterizes the cases for which $r'_{\text{ave}} = 0$.

PROPOSITION 21. *Suppose the blocked assignment a_f satisfies (25) in addition to the blocking equations (7), (8) and (9); i.e., a_f is an almost hypercubical blocking. Then $r'_{ave}(a_f) = 0$ iff*

$$q_{D,K}(\alpha) := (\alpha+1)^K \alpha^{(D-K)} - \left(\alpha + \frac{K-1}{D}\right)^D > 0. \tag{26}$$

Proof: Equation (11) shows that $r'_{\text{ave}}(a_f) = 0$ iff $\sum g = \left\lceil D\,(\prod g)^{\frac{1}{D}} \right\rceil$. The arithmetic-geometric mean inequality shows already that $\sum g \geq D\,(\prod g)^{\frac{1}{D}}$, sufficing it to show that $\sum g - 1 < D\,(\prod g)^{\frac{1}{D}}$. The result follows because $\sum g = D\alpha + K$ and $\prod g = (\alpha+1)^K \alpha^{D-K}$. ∎

The next result shows that, for any fixed D, almost hypercubical blocking is provably optimal for α sufficiently large.

PROPOSITION 22. *If a_i is the coefficient of α^i in $q_{D,K}$, then $a_D = 0$ and $a_{D-1} = 1$.*

Proof: By the binomial theorem,

$$a_i = \begin{cases} -\left[\dfrac{K-1}{D}\right]^{D-i} \binom{D}{i} & \text{if } i \in \{0,\ldots,D-K-1\} \\ \binom{K}{i-D+K} - \left[\dfrac{K-1}{D}\right]^{D-i} \binom{D}{i} & \text{if } i \in \{D-K,\ldots,D\}. \end{cases}$$

So $a_D = 1 - 1 = 0$ and $a_{D-1} = K - [(K-1)/D]D = 1$. ∎

The following technical lemma on polynomials will be useful for considering general α.

LEMMA 23. *Given a polynomial $q(\alpha) = a_n\alpha^n + \cdots + a_0$, where the coefficients are of the form*

$$a_n > 0, \qquad a_{n-1},\ldots,a_k \geq 0 \qquad \text{and} \qquad a_{k-1},\ldots,a_0 \leq 0. \tag{27}$$

Then $(\ \bar\alpha > 0 \ \text{ and } \ q(\bar\alpha) > 0\ ) \implies \forall\{\alpha \geq \bar\alpha\}\ q(\alpha) > 0.$

Proof: Clearly, if we restrict ourselves to $\alpha \geq \bar\alpha > 0$, $q(\alpha) > 0$ iff $r(\alpha) := q(\alpha)/\alpha^k > 0$. But, for

$$r(\alpha) = \underbrace{a_n\alpha^{n-k} + \cdots + a_{k+1}\alpha}_{s(\alpha)} + a_k + \underbrace{\frac{a_{k-1}}{\alpha} + \cdots + \frac{a_0}{\alpha^k}}_{t(\alpha)}$$

and $\alpha \geq \bar\alpha$, we have $0 < s(\bar\alpha) \leq s(\alpha)$ and $0 \geq t(\alpha) \geq t(\bar\alpha)$. Therefore $r(\alpha) = s(\alpha) + a_k + t(\alpha) \geq s(\bar\alpha) + a_k + t(\bar\alpha) = r(\bar\alpha) > 0$. ∎

We believe that $q_{D,K}$ satisfies the hypothesis of the preceding lemma, but have been unable to prove that result in general. Therefore we propose the following:

CONJECTURE 24. *The polynomial $q_{D,K}(\cdot)$ defined in (26) has coefficients of the form (27).*

However, for given D, it is possible to compute $q_{D,K}$ for each positive $K < D$ and check the conjecture. We have verified that it is true whenever $D \leq 12$ using the PARI calculator (see [1], [2]).

COROLLARY 25. *Suppose that conjecture 24 is true for a given pair of positive integers D and $K < D$. Then $r'_{ave}(a_f) = 0$ for all blocked assignments a_f corresponding to the pair D, K and all positive integers α iff $q_{D,K}(1) > 0$.*

Proof: Follows directly from proposition 21 and lemma 23. ∎

Using the PARI calculator, we have verified that $q_{D,K}(1) > 0$ for $0 < K < D \leq 11$. This proves that almost hypercubical blocking is optimal in dimensions $D \leq 11$. However, if $D = 12$, then $q_{D,K}(1) < 0$ for $K \in \{6, 7\}$. Thus we are left with some interesting open questions regarding optimal solutions for $D \geq 12$.

References

[1] C. Batut. Présentation du système PARI. *Actes des journées Mathématique et Informatique de Marseille-Luminy*, pages 1–20, 1988.

[2] C. Batut, D. Bernardi, H. Cohen, and M. Oliver. User's guide to PARI-GP. Private communication.

[3] E.F. Beckenbach and R. Bellman. *Inequalities.* Springer-Verlag, Berlin, 1961.

[4] D. Bitton, D. DeWitt, and C. Turbyfill. Benchmarking database systems: A systematic approach. In *Proceedings of the 1983 VLDB Conference*, October 1983.

[5] H. Boral, W. Alexander, L. Clay, G. Copeland, S. Danforth, M. Franklin, B. Hart, M. Smith, and P. Valduriez. Prototyping Bubba, a highly parallel database system. *IEEE Transactions on Knowledge and Data Engineering*, 2(1), March 1990.

[6] A. Brooke, D. Kendrick, and A. Meeraus. *GAMS—A User's Guide.* The Scientific Press, 1988.

[7] D. DeWitt, S. Ghandeharizadeh, D. Schneider, A. Bricker, H. Hsiao, and R. Rasmussen. The Gamma database machine project. *IEEE Transactions on Knowledge and Data Engineering*, 2(1), March 1990.

[8] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*, pages 60–62. W.H. Freeman and Company, New York, 1979.

[9] S. Ghandeharizadeh. *Physical Database Design in Multiprocessor Systems.* PhD thesis, University of Wisconsin - Madison, 1990.

[10] G. Graefe. Volcano: An extensible and parallel dataflow query processing system. Computer science technical report, Oregon Graduate Center, Beaverton, OR, June 1989.

[11] G.H. Hardy, J.E. Littlewood, and G. Polya. *Inequalities.* Cambridge, 1959.

[12] P. Helman. A family of NP-complete data aggregation problems. *Acta Informatica*, 26:485–499, 1989.

[13] H.L. Keng. *Introduction to Number Theory.* Springer-Verlag, 1982.

[14] M. Livny, S. Khoshafian, and H. Boral. Multi-disk management algorithms. In *Proceedings of the 1987 ACM SIGMETRICS Int'l Conf. on Measurement and Modeling of Computer Systems*, May 1987.

[15] D. Ries and R. Epstein. Evaluation of distribution criteria for distributed database systems. UCB/ERL Technical Report M78/22, UC Berkeley, May 1987.

[16] M. Stonebraker, D. Patterson, and J. Ousterhout. The design of XPRS. In *Proceedings of the 1988 VLDB Conference*, Los Angeles, CA, September 1988.

[17] Tandem Performance Group. A benchmark non-stop SQL on the debit credit transaction. In *Proceedings of the 1988 SIGMOND Conference*, Chicago, IL, June 1988.

[18] Teradata Corp. *DBC/1012 Data Base Computer System Manual*, November 1985. Teredata Corp. Document No. C10-0001-02, Release 2.0.