Block Access Estimation for Clustered Data Using a Finite LRU Buffer

Fabio Grandi and Maria Rita Scalas

Abstract— **Data access cost evaluation is fundamental in the design and management of database systems. When some data items have duplicates, a clustering effect which can heavily influence access costs is observed. The availability of a finite amount of buffer memory in real systems has an even more dramatic impact. In this paper a comprehensive cost model for clustered data retrieval by an index using a finite buffer is presented. Our approach combines and extends previous models based either on finite buffer or on uniform data clustering assumptions. The computational cost of the formulas we propose in this work is independent of the data size or of the query cardinality and need only a single statistics per search key, the *clustering factor*, to be maintained by the system. The predictive power and the accuracy of the model are shown in comparison with actual costs resulting from simulations.**

Index Terms—**Block accesses, buffer management, databases, data clustering, indexed access, performance evaluation, physical design, query optimization, relational database.**

I. Introduction and Background

Analytic estimation of the number of blocks accessed by a query is a key problem in the realm of databases. Performance evaluation of database systems is strictly dependent on the availability of reliable and accurate I/O cost models which are based on block access estimation. The application field of these models ranges from physical database design, for the choice of indexes ([10], for instance), to query optimization at run-time, for access path selection ([17], for instance).

The applicability of cost models relies on their capability to accurately capture the behavior of the processes they describe. Such a constraint is even stronger in the case of design and management of high-performance systems. Moreover, the usefulness of a cost model is related to its economy, with respect to the computational burden required, the memory used for parameter storage and the related bookkeeping overhead.

In this paper we consider the evaluation of the I/O cost paid to access data by an index in terms of the number of accesses to disk pages (blocks). We consider the problem in a relational perspective and use, for convenience, the related terminology (namely "relation" instead of "file," "tuple" instead of "record," "attribute" instead of "field," etc.), although the problem is general and the results can be applied in a broad class of database management and file systems. A summary of the symbols used is reported in Table I. We use the term "key" to denote the search attribute on which an index has been built, and is used for access by queries. Indexes are assumed to be B^+-trees or similar [9]. In this framework, the *cost estimation problem for data access by an index* can be stated as follows:

> Estimate the number FP of page fetches needed in order to retrieve all the tuples matching a given number HK of key values, using an index built on that key and with B pages of the buffer pool available for the relation[1]

Early access cost models were based on simplifying assumptions which can be summarized as follows:

- *Total uniformity*: the relation has a constant number of tuples per page and each tuple has the same probability to be referenced by queries.
- *Unlimited buffer*: the pages accessed can be kept in main memory till the end of the query and further references to them do not give rise to additional I/O costs.

The first hypothesis leads to inaccurate estimates the more the data distribution over the key space and over the pages differs from the uniform distribution. The second hypothesis is obviously never met in real systems and, in very common situations, it can lead to such strong cost underestimations that they are practically useless.[2] More sophisticated models proposed so far loosened one of the two assumptions, but none of them both. The aim of this work is to abandon both of them, providing a cost model which can be applied to the retrieval of data by an index using a *finite buffer*, with the *total uniformity* assumption abandoned in favor of the following one:

- *Uniform clustering*: the relation has a constant number of distinct key values per page and each key value has the same probability to be referenced by queries.

Usage and maintenance of cost models based on *uniform clustering* are not expensive, because they are based on a single parameter per attibute: the *clustering factor* [3], [13], which could easily be embedded in systems already in use. Cost models based on this parameter give results that are encouraging for their accuracy in a wide range of situations. *Notice that the term* clustering *is used to indicate the presence of more than one tuple with the same key value on the same page.* Total clustering *means all the tuples with the same*

Manuscript received January 2, 1992; revised November 23, 1992. Recommended by Matthias Jarke.

The authors are with Centro di studio per l'Interazione Operatore–Calcolatore, Consiglio Nazionale delle Richerche and Dipartimento di Elettronica Informatica e Sistemistica, Università di Bologna, I-40136 Bologna, Italy

IEEE Log Number 9208631.

[1] We disregard the buffer space and the I/O traffic devoted to index management.

[2] Notice that the finite buffer assumption is required only when retrieved tuples have to be in sequential order of the keys (e.g., during a join operation). Otherwise, if the *page* requests can be batched and the duplicate requests are eliminated, cost models assuming an unlimited buffer are adequate.

TABLE I
THE SYMBOLS USED

B	the dimension of the buffer in pages
R	a given relation
NT	number of tuples in R
NP	number of pages of R
NK	number of distinct key values in R
TP	average number of tuples per page
KP	average number of distinct key values per page
DK	average duplication of key values
CF	average clustering factor of the key attribute
HT	number of tuples retrieved by a query
HK	number of keys retrieved by a query
HP	number of pages hit
FP	number of pages fetched
	with the constraints: $NT = NP \cdot TP = NK \cdot DK, KP \cdot CF = TP$

TID	A_1	A_2	A_3	A_4
1.1	a	b	d	c
1.2	e	e	d	c
1.3	b	a	b	c
1.4	d	a	d	c
1.5	f	e	b	c
1.6	c	b	b	c
2.1	f	c	a	e
2.2	d	c	f	e
2.3	b	f	f	e
2.4	e	d	a	e
2.5	a	f	a	e
2.6	c	d	f	e
3.1	c	c	b	a
3.2	a	a	c	a
3.3	f	d	b	a
3.4	e	c	b	a
3.5	d	d	c	a
3.6	b	a	b	a

TID	A_1	A_2	A_3	A_4
4.1	a	e	e	f
4.2	b	f	a	f
4.3	d	b	e	f
4.4	f	b	a	f
4.5	c	e	e	f
4.6	e	f	a	f
5.1	e	f	c	d
5.2	b	b	c	d
5.3	f	a	c	d
5.4	d	b	f	d
5.5	c	a	f	d
5.6	a	f	f	d
6.1	c	e	e	b
6.2	b	d	d	b
6.3	e	d	d	b
6.4	f	c	d	b
6.5	a	c	e	b
6.6	d	e	e	b

Fig. 1. The figure shows the six pages of a sample relation. The attributes A_1, A_2, A_3, A_4 assume values uniformly distributed over the common domain {a,b,c,d,e,f} ($DK_{A_1} = DK_{A_2} = DK_{A_3} = DK_{A_4} = 6$). Each attribute satisfies the *uniform clustering* model for queries uniformly referencing key values ($CF_{A_1} = 1$, $CF_{A_2} = 2$, $CF_{A_3} = 3$, $CF_{A_4} = 6$). In particular, A_1 fits *total uniformity* and A_4 fits *total clustering*. The column TID (not belonging to the relation) contains the Tuple IDentifiers in the format: Page IDentifier . offset within the page.

key value are consecutive. Ordering *is a special case of total clustering in that the key values are sorted* (See Fig. 1).

More sophisticated and costly models [5], [18] based on complete page access profiles [14] are actually needed only in extreme situations, since they are the only way to guarantee accuracy in the presence of highly skewed distributions. However, cost models for queries on one attribute, based on *uniform clustering*, are only $O(1)$ with respect to NP and NK both in time and in space, whereas more accurate models are at least $O(NP)$ or $O(NK)$ in time or in space.

The estimation problems dealt with in this paper are often related to the combinatorial analysis of the selection of *distinct* items (e.g., tuples, keys, key occurrences) from a "paged" population (i.e., partitioned in granules). This analysis can be based on exact models, enforcing selection without replacement, or on approximated models, allowing selection with replacement. In Section II the approximation problem is reexamined in a general framework, introducing the concept of *feasibility* of a replacement-based approximation, which will be applied throughout the paper. In Section III we present our generalized I/O cost model for data access via an index, which takes into account data clustering and finite buffer space. Section IV is devoted to the description and discussion of the simulation results based on the cost model proposed, showing the relevance of the precision and the improvement introduced with respect to the previous models which consider the two aspects (clustering and finite buffer) separately.

The rest of this section provides an introductory review of the access estimation formulas previously proposed, which are extended by the model presented in this paper. They are classified with respect to their applicability to the cost estimation problem.

A. Cost Models for Total Uniformity and Unlimited Buffer

The exact formula for estimating the number HP of pages *hit* by a query retrieving HT distinct tuples, under the *total uniformity* assumption, is

$$HP(HT) = NP\left[1 - \frac{\binom{NT-TP}{HT}}{\binom{NT}{HT}}\right] \qquad (1)$$

which was independently derived by Yao [21] and Waters [20]. Considering the general access-by-an-index cost estimation problem, formula (1) exactly applies, under the unlimited buffer assumption, to the case of ***unique key***, if we let $HT = HK$. In the case of duplication, it also applies with $HT = HK \cdot DK$ and the assumption of ***total uniformity*** implies random placement of tuples on the pages and uniform distribution of key values both in the tuples and in the query references.

Two other formulas have been proposed to solve the same problem. The first one:

$$HP(HT) = NP\left[1 - \left(1 - \frac{1}{NP}\right)^{HT}\right] \qquad (2)$$

was independently derived by Cárdenas [4] and Karayannis–Waters [20] and gives the exact expected cost for the retrieval of HT tuples *not necessarily distinct*. The second one:

$$HP(HT) = NP\left[1 - \left(1 - \frac{HT}{NT}\right)^{TP}\right] \qquad (3)$$

was independently derived by Waters [20] and Palvia–March [15]. Both the above formulas can be used as good approximations of Yao's formula (1) as discussed, for instance, in [15], [19], [20], [21] and in Section II of this paper.

B. Cost Models for Uniform Clustering and Unlimited Buffer

The average *clustering factor* of an attribute was originally introduced in [3] as

$$CF = \frac{NT}{NPID} \qquad (4)$$

where $NPID$ represents the total number of different page identifiers (*PID*'s) which can be found in the leaves of an index built on that attribute. This parameter takes into account the actual placement of the different key values on the pages, since it represents the average number of tuples with the same key value on a page. The average is taken over the key values and over the pages. If we define as *cluster* the set of occurrences of the same key on a page, the CF represents the average dimension of a cluster.

Assuming a constant number of clusters per page and clusters to be uniformly distributed over the pages, the following formula:

$$HP(1) = \frac{NPID}{NK} \tag{5}$$

was proposed in [13] to estimate the average number of pages accessed to retrieve all the tuples matching a *single* key value via an index. Using the CF definition (4) and the identity $NT = NK \cdot DK$, formula (5) can also be rewritten as

$$HP(1) = \frac{DK}{CF}. \tag{6}$$

The clustering factor was also used in [2] in order to predict the optimal number of contiguous pages to be simultaneously transferred from disk. In that paper also a naive extension of formula (5) to cover the general case $HK \geq 1$ was proposed as

$$HP(HK) = HK \cdot HP(1). \tag{7}$$

This formula, which gives the total number of page references generated by the query, clearly overestimates the expected cost since it does not consider interleaving of different hit clusters within hit pages. This formula represents a first order approximation of the exact one [like $HT \cdot TP/NT$ [20] represents a first order approximation of formulas (1), (2), and (3)] and is valid only for a very low number of key values hit.

A more accurate cost model for the retrieval of all the tuples matching HK keys via an index is given by Ciaccia–Scalas' formula:

$$HP(HK) = NP\left[1 - \frac{\binom{NK - TP/CF}{HK}}{\binom{NK}{HK}}\right] \tag{8}$$

proposed in [6], which gives the expected number of accessed pages, assuming a constant number of tuples per page and a uniform distribution of key values on the pages and in the query references in addition to the *unlimited buffer* assumption.

An approximated formula for the same problem:

$$HP(HK) = NP\left[1 - \left(1 - \frac{HP(1)}{NP}\right)^{HK}\right] \tag{9}$$

was independently derived by Bonfatti–Maio–Spadoni–Tiberio [3] and Ciaccia[3] [7]. This formula provides exact expectations if the HK key values are *not necessarily distinct*.

In Section II-B we show how these models can be used under the more general hypothesis of *uniform clustering* (and *unlimited buffer*). The *uniform clustering* assumption is less restrictive than the assumptions on which formulas (8) or (9) are based, since it does not require a constant number of tuples per page.

C. Cost Models for Total Uniformity and Finite Buffer

All the formulas so far mentioned implicitly assume the availability of an infinite amount of main memory. In fact, they provide an estimation of the number of *block hits*, which is only the lower bound (and often a very poor estimate) of the number of *block fetches* required to answer the query in a real environment. Accessed pages brought by the system into the buffer are indeed subject to the LRU replacement policy, so that, if they are further referenced by the query, they may actually no longer be present in the buffer after the last reference and may need to be repeatedly fetched.

A formula which considers a finite buffer space is Palvia's formula that appeared in [16]:

$$FP(HT) = \begin{cases} HP(HT) & \text{if } HP(HT) \leq B \\ B + A_i & \text{if } HP(HT) > B \end{cases} \tag{10}$$

where $HP(HT)$ is computed with Waters' formula (3). For the term A_i—representing the number of pages to be accessed after the buffer is full—three approximations have been proposed:

$$A_1 = (HT - \overline{HT})\frac{NT - B \cdot TP}{NT - \overline{HT}}$$

$$A_2 = (HT - \overline{HT})\frac{NT - B \cdot TP - (HT - \overline{HT})/2}{NT - \overline{HT} - (HT - \overline{HT})/2}$$

$$A_3 = (HT - \overline{HT}) \times \frac{NT - B \cdot TP - (HT - \overline{HT})/2 + Q_1 - \overline{HT}}{NT - \overline{HT} - (HT - \overline{HT})/2}$$

where

$$\overline{HT} = HP^{-1}(B) = NT\left[1 - \left(1 - \frac{B}{NP}\right)^{1/TP}\right]$$

$$Q_1 = (\overline{HT} + B \cdot HT/HP(HT))/2.$$

These aproximations are based on three different degrees in introducing *selection without replacement* in the evaluation of the probability that a referenced page may already be found in the buffer. Palvia's formula (10) is based on the *total uniformity* assumption and can be used to evaluate the access cost by an index, if the key is *unique* ($DK = 1$), letting $HT = HK$. In the general case, with $HT = HK \cdot DK$ when $DK \geq 1$, it can provide an overestimation of the actual I/O cost, since it does not take into account the actual clustering of data. As a matter of fact, also when the *total uniformity* hypothesis is met, duplication causes more than one tuple with the same key value to be placed on the same page. We call this phenomenon *natural clustering induced by uniformity*. Formula (10) provides the same results either if the HT tuples

[3] In the original paper, the term $TP/NK \cdot CF$ replaces $HP(1)/NP$. The equivalence can be trivially shown by means of (6) and identity $DK/NP = TP/NK$.

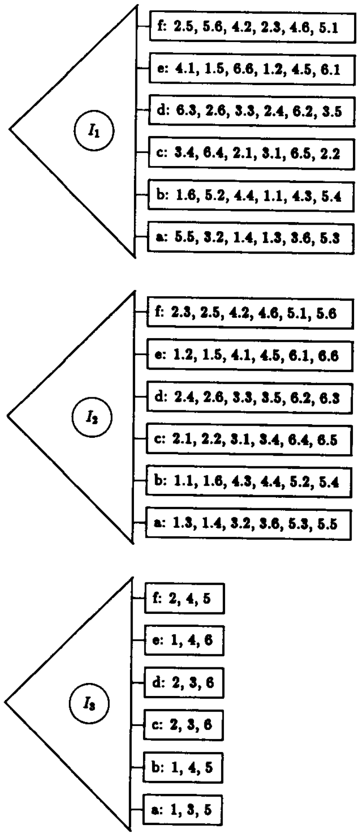

I_1
f: 2.5, 5.6, 4.2, 2.3, 4.6, 5.1
e: 4.1, 1.5, 6.6, 1.2, 4.5, 6.1
d: 6.3, 2.6, 3.3, 2.4, 6.2, 3.5
c: 3.4, 6.4, 2.1, 3.1, 6.5, 2.2
b: 1.6, 5.2, 4.4, 1.1, 4.3, 5.4
a: 5.5, 3.2, 1.4, 1.3, 3.6, 5.3
I_2
f: 2.3, 2.5, 4.2, 4.6, 5.1, 5.6
e: 1.2, 1.5, 4.1, 4.5, 6.1, 6.6
d: 2.4, 2.6, 3.3, 3.5, 6.2, 6.3
c: 2.1, 2.2, 3.1, 3.4, 6.4, 6.5
b: 1.1, 1.6, 4.3, 4.4, 5.2, 5.4
a: 1.3, 1.4, 3.2, 3.6, 5.3, 5.5
I_3
f: 2, 4, 5
e: 1, 4, 6
d: 2, 3, 6
c: 2, 3, 6
b: 1, 4, 5
a: 1, 3, 5

Fig. 2. Unclustered index leaf organizations: I_1, with unsorted TID's; I_2 with sorted TID's; I_3 with PID's. The indexes are built on the attribute A_2 of the relation in Fig. 1.

match the same key value or if they match different key values, because it deals only with page references in tuple searching. *Natural clustering* implies that multiple references to the same page may be generated in retrieving all the occurrences of a single key value. If the index leaves contain tuple identifiers (*TID*'s, composed of a PID and an offset) and TID groups corresponding to single key values are sorted by their PID component, or if the index leaves contain only the PID's [3], then multiple references to the same page in retrieving the occurrences of a single key are processed together and require a single page access (See Fig. 2). Formula (10) considers all the referenced pages to be distinct and subject to LRU replacement in the buffer pool. Therefore, formula (10) is accurate for an access by an index, when $DK > 1$, only if the index is *unclustered*, organized with TID's in the leaves and with TID groups not sorted. Only in this case can the same pages really be accessed, out of the buffer pool, more than once, also when different tuples with the same key value are retrieved.

Other access cost models proposed so far take into account *natural clustering induced by uniformity*. Such models base their cost evaluation on the number of page references generated by the use of the index rather than on the number of referenced tuples like Palvia's formula (10).

A first cost model of this kind has been embedded in the optimizer module of the relational DBMS System R [1]:

$$FP(HK) = \begin{cases} \min\{HK\lceil HP(DK)\rceil, NP\} & \text{if } NP \leq B \\ HK\lceil HP(DK)\rceil & \text{if } NP > B \end{cases} \tag{11}$$

where the term $HP(DK)$ is computed using Cárdenas' formula (2) (where DK is the argument) and thus represents the number of pages spanned by a single key value under the *total uniformity* assumption. Formula (11) is pessimistic, since it assumes a number of page fetches equal to the number of page references generated by the query, which is estimated as the product of the number of retrieved key values times the round up number of pages spanned by a single key value. In other words, formula (11) neglects the possibility of a key value being found on a page already referenced by another key value and that such a page can still be found in the buffer pool.

Two more precise cost models have been presented by Mackert and Lohman. The Mackert–Lohman's first formula:

$$FP(HK) = \begin{cases} \min\{HK \cdot NP(1-q), NP\} & \text{if } NP \leq B \\ B + [HK \cdot NP(1-q) - B]\frac{NP-B}{B} & \text{if } NP > B \end{cases} \tag{12}$$

where

$$q = \begin{cases} (1 - 1/NP)^{DK} & \text{if } DK \leq TP \\ (1 - 1/NK)^{TP} & \text{if } TP \leq DK \end{cases} \tag{13}$$

was presented in [11]. It is still based on the number of references generated by the index, but a nonnull probability $(NP - B)/NP$ that a page be found in the buffer is considered. The term $NP(1 - q)$ represents the expected value of the number of pages spanned by one key value, under the *total uniformity* assumption, which is more accurate than the approximation $HP(DK)$ used by the System R cost model (11). The Mackert–Lohman's second formula:

$$FP(HK) = \begin{cases} NP(1 - q^{HK}) \\ \qquad \text{if } HK \leq \overline{HK} \\ NP\left(1 - q^{\overline{HK}}\right) + (HK - \overline{HK})NP(1-q)q^{\overline{HK}} \\ \qquad \text{if } \overline{HK} < HK \leq NK \end{cases} \tag{14}$$

where

$$\overline{HK} = \max\{HK \in \{0, \cdots, NK\} | NP(1 - q^{HK}) \leq B\} \tag{15}$$

was presented in [12]. The value of q is the same value (13) as that used in (12). This formula is the starting point of our proposal and therefore will be discussed in detail in Section III. The main difference between Mackert–Lohman's two formulas is that the second one takes into account the possibility that a page can be re-referenced without being refetched even when the buffer is not yet full, whereas the first one does not. Two less costly approximations of formula (14) were also proposed in [12].

II. Feasibility of Replacement-Based Approximations

Consider the classical situation of b distinguishable balls thrown into d distinguishable buckets of finite capacity c. We can think of a bucket as consisting of c distinguishable boxes into each of which fits exactly one ball. Thus we have a number $a = cd$ of boxes in which a ball can be found after the throw. The usual assumption is that all the boxes have the same probability of being hit by a ball and all the outcomes of the throw are equifrequent.

In particular, we are interested in the probability $\wp$ that a given bucket is not hit by any ball. In order to evaluate $\wp$, we can consider the situation after the throw as if it were produced by two different experiments involving selection of boxes *without replacement*:

- *Primal Experiment*: Selection of the b boxes to be assigned to the thrown balls.
- *Dual Experiment*: Selection of the c boxes to be assigned to the given bucket.

In the *primal* experiment, $\wp$ can be evaluated as the probability that the selected boxes do not belong to the given bucket, yielding the well known expression

$$\wp = \frac{\binom{a-c}{b}}{\binom{a}{b}}. \tag{16}$$

In the *dual* experiment, $\wp$ can be evaluated as the probability that the selected boxes do not contain any ball, yielding

$$\wp = \frac{\binom{a-b}{c}}{\binom{a}{c}}. \tag{17}$$

It can be easily shown that the two experiments, enforcing *nonreplacement* of boxes, give rise to the same value of $\wp$. Moreover, since formulas (16) and (17) can be evaluated as

$$\prod_{j=1}^{b} \frac{a-c-j+1}{a-j+1} = \prod_{j=1}^{c} \frac{a-b-j+1}{a-j+1},$$

$\wp$ can be computed in $O(\min\{b, c\})$ operations using the product which has the lowest number of terms.

Instead of (16) or (17) less expensive approximations based on experiments involving selection of boxes *with replacement* could be preferable. In this case, the *primal* and the *dual* experiments no longer provide the same results, since they represent *different approximated models* of reality.

In the *primal replacement* experiment we allow the b boxes hit by balls to be selected *with replacement*. All the selected boxes have the same probability c/a of belonging to a given bucket, which does not depend on the number of boxes already selected during the experiment. Therefore, we can use for $\wp$ the approximation

$$\wp \simeq \left(1 - \frac{c}{a}\right)^{b}. \tag{18}$$

The *primal replacement* experiment allows the same box to be selected even b times. Since there is a total number of c boxes in a bucket, one box of the given bucket cannot be selected in any case more than c times in assigning boxes to balls, unless we allow the bucket capacity to be exceeded. Therefore, we should consider (18) as a *feasible* replacement approximation of $\wp$ only if $b \leq c$, and *unfeasible* otherwise.

In the *dual replacement* experiment we allow the c boxes composing the given bucket to be selected *with replacement*. All the selected boxes have the same probability b/a of containing a ball, which does not depend on the number of boxes already selected (i.e., assigned to the bucket) during the experiment. Therefore, we can use for $\wp$ the approximation

$$\wp \simeq \left(1 - \frac{b}{a}\right)^{c}. \tag{19}$$

The *dual replacement* experiment allows the same box to be selected even c times. Since there is a total number of b boxes containing a ball, a box containing a ball cannot be selected in any case more than b times in assigning boxes to the bucket, unless we allow the number of boxes hit in a bucket to be greater than the number of balls thrown. Therefore, we should consider (19) as a *feasible* replacement approximation of $\wp$ only if $c \leq b$, and *unfeasible* otherwise.

Hence, a globally *feasible replacement* approximation for the probability $\wp$ is given by

$$\frac{\binom{a-c}{b}}{\binom{a}{b}} = \frac{\binom{a-b}{c}}{\binom{a}{c}} \simeq \begin{cases} (1-c/a)^{b} & \text{if } b \leq c \\ (1-b/a)^{c} & \text{if } c \leq b \end{cases}$$
$$= \left(1 - \frac{\max\{b,c\}}{a}\right)^{\min\{b,c\}} \tag{20}$$

which has a complexity $O(1)$ with respect to a, b, and c. Another interesting feature of the *replacement approximations* is that linear functions of the exact ratio (16) or (17), regarded as a dependence on b or c, cannot be solved in function of their argument in closed form, whereas their approximations (18) and (19) can easily be inverted. This peculiarity will be used throughout the paper.

A. Applications of the Feasible Replacement Approximation

Referring to (20), we can assume, for instance, that the buckets represent pages of a relation R (thus $d = NP$) and the boxes to represent tuples (namely $c = TP$ and $a = NT$) which are in this case all distinct ($NK = NT$, i.e., no key values have duplicates). The throw of the balls corresponds to a query issued on R, in which $b = HT$ distinct tuples have to be retrieved. Using (16) or (17), we can express the exact probability that a given page is not hit under the *total uniformity* assumption. If we use such a probability to compute the expected number of blocks hit, Yao's formula (1) is derived. Considering, on the other hand, the replacement approximations (18) or (19) we can derive Cárdenas' (2) or Waters' (3) formulas, respectively. However, according to our definition of *feasibility*, the only globally *feasible* replacement approximation of (1) must be derived using (20):

$$HP(HT) \simeq NP\left[1-\left(1-\frac{\max\{HT,TP\}}{NT}\right)^{\min\{HT,TP\}}\right]. \qquad (21)$$

In particular, Cárdenas' formula is a feasible approximation if $HT \leq TP$ whereas Waters' formula is a feasible approximation for $HT \geq TP$. Notice that although Cárdenas' formula is usually adopted as an approximation of Yao's formula, the only feasible approximation is given in *most* cases by Waters' formula, since HT may range from 1 to NT whereas generally $TP \ll NT$ and thus HT will be frequently greater than the feasibility upper bound TP. Formula (21) is equivalent to the formula proposed by Wang–Wiederhold–Sagalowicz [19]: observing that both Cárdenas' and Waters' formulas give underestimates of Yao's formula, they proposed to take the maximum between the two, which easily reduces to (21). For mnemonic reasons, we will denote the approximations *á la* Cárdenas or *á la* Waters the *primal* or *dual feasible replacement* approximation, respectively.

Another example of the application of the *feasible replacement* approximation can be given in considering the random placement of uniformly distributed key values on pages of constant capacity, when key values have duplicates. The probability that a given page does not contain a given key value can be evaluated as the probability that a given page is not accessed in placing (or retrieving) all the DK occurrences of the key, with individual occurrences of keys considered as distinct tuples. This probability is therefore exactly [12]

$$\frac{\binom{NT-TP}{DK}}{\binom{NT}{DK}} = \frac{\binom{NT-DK}{TP}}{\binom{NT}{TP}} \qquad (22)$$

whose *feasible replacement* approximation is the q (13) used in the Mackert–Lohman' formulas (12) and (14).

B. Unlimited Buffer and Uniform Clustering Models Revisited

The *uniform clustering* assumption, as defined in the Introduction, implies that the number of different key values (clusters) per page is constant and, thus, equals its average value KP. In this case, we can exactly evaluate the probability that a given page is not accessed in retrieving HK distinct key values as

$$\frac{\binom{NK-KP}{HK}}{\binom{NK}{HK}}$$

where the denominator expresses the total number of query instances with cardinality HK, while the numerator represents the number of them not referencing the considered page. Therefore, the expected number of pages hit under the *uniform clustering* assumption can be computed as

$$HP(HK) = NP\left[1-\frac{\binom{NK-KP}{HK}}{\binom{NK}{HK}}\right]. \qquad (23)$$

Notice that the derivation of (23) rigorously requires neither uniform distribution of key values in the tuples, nor a constant number of tuples per page. However, if the number of tuples per page is constant, *uniform clustering* implies that also the *clustering factor* is constant, owing to

$$KP = \frac{TP}{CF}. \qquad (24)$$

Since identity (24) implies the equivalence of formulae (8) and (23), if TP and CF are both constant, then Ciaccia–Scalas' formula (8) exactly applies to the cost estimation problem assuming *uniform clustering* and *unlimited buffer*.

When the number of different key values per page is not constant, formula (23) provides in many cases a good approximation if the average value KP is used, even if the derivation of (23) can no longer be soundly justified from a probabilistic point of view (e.g., higher order moments of the number of different key values per page should be considered). Formula (8) also provides the same approximation thanks to identity (24) with the average values. The applicability of formula (23) with the average KP only requires that the underlying distributions—frequencies of key values and their placement—are not highly skewed, as happens when different nonuniformities partially compensate each other rather than increase. Encouraging results about accuracy of cost models based on (average) *uniform clustering* have already been obtained in [7]. The simulations reported in Section IV also confirm the quality of the approximation, which will be further investigated by the authors.

The *feasible replacement* approximation of (23) is

$$HP(HK) \simeq NP\left[1-\left(1-\frac{\max\{HK,KP\}}{NK}\right)^{\min\{HK,KP\}}\right]. \qquad (25)$$

Using identity (24) formula (25) becomes

$$HP(HK) \simeq NP\left[1-\left(1-\frac{\max\{HK,TP/CF\}}{NK}\right)^{\min\{HK,TP/CF\}}\right]. \qquad (26)$$

Notice that the approximation *á la* Cárdenas, feasible for $HK \leq TP/CF$, is the original form of Ciaccia's formula (9), whereas the approximation *á la* Waters, feasible for $HK \geq TP/CF$, is

$$HP(HK) \simeq NP\left[1-\left(1-\frac{HK}{NK}\right)^{TP/CF}\right] \qquad (27)$$

and has not, to the authors' knowledge, yet been proposed.

In this framework, *total uniformity* can be considered as a particular case of *uniform clustering*. First, the random placement of tuples in the presence of duplication of key values gives rise to what was defined *natural clustering* in the Introduction. It is sufficient that $DK > 1$ in order to have a nonnull probability that more than one tuple matching the same key value be placed on the same page and, thus, clusters be formed, even though the random placement causes maximal scattering of the tuples with the same key value over the pages. For instance, if $DK \simeq 2 \cdot NP$, there is a high probability that every page contains *two* tuples matching a common key. In general, *total uniformity* implies a clustering factor usually greater than the absolute lower bound $CF = 1$ which is reached only if $DK = 1$. Secondly, formulas (23) and (25) [or the equivalent ones (8) and (26) with the CF] can be used with the average value of KP induced by uniformity. This value does not depend on the page, as can be seen from its expected value

$$KP = NK(1-q) \tag{28}$$

where q is the probability [exact (22) or *feasible* approximation (13)] that a given key value is not found on a given page, which depends neither on the page nor on the key.

Therefore, *total uniformity* is a sufficient but not necessary condition for the applicability of a model based on *uniform clustering. As a matter of fact, if the number of tuples per page is constant and the key values are uniformly distributed in the tuples, the* uniform clustering *assumption is verified by a wide class of placements of tuples over the pages, ranging from the random placement to the case of sorted attribute.*

In the case of *total uniformity*, we can use the KP value (28) in order to estimate page hits with formula (23), obtaining

$$HP(HK) = NP\left[1 - \frac{\binom{NK\,q}{HK}}{\binom{NK}{HK}}\right] \tag{29}$$

and the replacement-based approximation

$$HP(HK) \simeq NP \times \left[1 - \left(1 - \frac{\max\{HK, NK(1-q)\}}{NK}\right)^{\min\{HK,NK(1-q)\}}\right]. \tag{30}$$

In the next section we show how Mackert–Lohman's second formula (14) can easily be understood by means of this equation.

III. A Solution to the Cost Estimation Problem

The purpose of this work is the derivation of a correct solution to the problem of I/O cost estimation for data access by an index in the presence of clustering and using a finite buffer space. In this section we present a comprehensive cost model which provides such a solution. Our model is based on an extension of Mackert–Lohman's formula (14) which takes into account buffer finiteness and *natural clustering induced by uniformity*. The extension is obtained by generalizing the original model to any actual value of the clustering factor. A more detailed characterization of the expected cost after the buffer is full and of the instant in which the buffer becomes full up are also provided. This detailed characterization is necessary since the introduction of the general clustering factor in the model makes it more sensitive to the estimation of this cost and instant. Moreover, simplifications are introduced in the generalized model in order to derive formulas which are $O(1)$ with respect to the problem dimension. Two different versions of the simplified cost model are finally proposed.

Let us extract from (30) the approximation *á la* Cárdenas of equation (29):

$$HP(HK) \simeq NP(1 - q^{HK}) \tag{31}$$

which takes into account the *natural clustering induced by uniformity* if (13) is used for q. By means of (31), we can rewrite Mackert–Lohman's formula (14) as

$$FP(HK) = \begin{cases} HP(HK) & \text{if } HK \le \overline{HK} \\ HP(\overline{HK}) + (HK - \overline{HK})HP(1)q^{\overline{HK}} & \\ \qquad\qquad \text{if } \overline{HK} < HK \le NK \end{cases} \tag{32}$$

with

$$\overline{HK} = \max\{HK \in \{0, \cdots, NK\} | HP(HK) \le B\}.$$

The meaning of (32) can be explained as follows:

- $HP(HK)$ represents an approximation of the number of pages containing HK distinct key values;
- if $HP(HK) \le B$ then all the accessed pages fit in the buffer without being replaced during the query execution. In this case $FP(HK) = HP(HK)$;
- if $HP(HK) > B$ then some of the accessed pages could be fetched more than once. $\overline{HK}$ is just the number of keys whose occurrences can *all* be always retrieved without exceeding the buffer capacity, thus without forcing blocks out of the buffer;
- $HP(\overline{HK})$ is the number of fetches required to read the first $\overline{HK}$ keys (in the *transient* during which the buffer is filled with $HP(\overline{HK})$ pages);
- $HK - \overline{HK}$ is the number of key values left to be retrieved (in the *steady-state* during which the buffer is already full with $HP(\overline{HK})$ pages);
- $HP(1)$ is the number of pages spanned by the occurrences of one key value (no page can be requested twice in retrieving all the occurrences of one key if we are using a PID-based index or a TID-based index with sorted TID groups);
- $(HK - \overline{HK})HP(1)$ is the total number of page requests issued during the *steady-state* (successive requests of the same page can be issued in retrieving occurrences of different key values);
- $HP(\overline{HK})/NP$ is the probability that any page of the relation is found in the buffer during the *steady-state* (*hit-in-buffer* probability), assuming selection of key values with replacement;
- $q^{\overline{HK}} = (1 - HP(\overline{HK})/NP)$ is the probability that a given page of the relation is *not* found in the buffer during the *steady-state*;

- $(HK - \overline{HK})HP(1)q^{\overline{HK}}$ is therefore the expected number of fetches required to complete the retrieval (namely page requests which do not find the page already in the buffer).

Two related remarks can be made. First, if $B \geq NP$, the whole relation fits into the buffer and thus $\overline{HK} = HK$ for any HK. In all the other cases, $\overline{HK}$ must be calculated from the definition. In [12] the iterative calculation was pointed out as one of the drawbacks of formula (32). It can be avoided by solving for $\overline{HK}$ the inequalities

$$HP(\overline{HK}) = NP\left(1 - q^{\overline{HK}}\right) \leq B$$
$$HP(\overline{HK}+1) = NP\left(1 - q^{\overline{HK}+1}\right) > B$$

with the constraint that $\overline{HK}$ is an integer, yielding

$$\begin{aligned}\overline{HK} &= \lfloor HP^{-1}(B) \rfloor \\ &= \left\lfloor \log_q \left(1 - \frac{B}{NP}\right) \right\rfloor = \left\lfloor \frac{\log(1 - B/NP)}{\log q} \right\rfloor.\end{aligned}$$

Secondly, when $HP(\overline{HK}) < B$ strictly, formula (32) is conservative, since the hit-in-buffer probability is computed with a *reduced* buffer capacity (i.e., $HP(\overline{HK})$), whereas the buffer management policy is able to use the whole buffer of B pages. However, this fact is counterbalanced by the optimistic assumption of a constant hit-in-buffer probability $HP(\overline{HK})/NP$, which does not take into account that the tuples already retrieved can no longer be referenced. Both these facts have been considered by Palvia for the nonclustering case (*total uniformity* with $DK = 1$). This author took into account the correct buffer capacity and *transient* length and, by the terms A_i in the formula (10), tried to evaluate a correct hit-in-buffer probability. In Mackert–Lohman's first formula (12) the correct buffer capacity is used to express the hit-in-buffer probability B/NP, which is the exact value for key selection *with replacement* and gives rise to an optimistic cost estimation, whereas a conservative estimation is adopted for the *transient* length, which gives rise to a pessimistic cost estimation. The two approximations still try to counterbalance each other. In our model these two aspects are not neglected and a globally complete characterization of the query behavior is introduced. We actually need more accurate estimations for both the *transient* length and the *steady-state* hit-in-buffer probability in order to obtain a cost model whose accuracy be uniform with respect to different values of the clustering factor. In particular (for long queries), the expected cost is more sensitive to the estimation of the hit-in-buffer probability when CF is small, because the *transient* will be short in this case and its contribution to the global cost lower, whereas the expected cost is very sensitive also to an accurate estimation of the *transient* length when the CF is high.

In the following, we no longer consider the case $B \geq NP$, which is equivalent to dealing with an infinite buffer and therefore leads to $FP(HK) = HP(HK)$.

A. Fixing the Useful Buffer Capacity and Transient Length

In order to take into account the whole buffer capacity B, an improvement of the formula (32) is given by

$$FP(HK) = \begin{cases} HP(HK) & \text{if } HK \leq \overline{HK}' \text{ (or } B \geq NP) \\ B + (HK - \overline{HK}')HP(1)(1 - \frac{B}{NP}) & \\ \qquad\qquad \text{if } \overline{HK}' < HK \leq NK & \end{cases} \tag{33}$$

where $\overline{HK}'$ is the real number of key values required to fill the buffer, and the hit-in-buffer probability is evaluated as B/NP. Notice that in a replacement approximation context, the pages in the buffer should be considered as a truly *random subset* of the NP pages of the relation due to the LRU replacement policy, regardless of tuples previously referenced in them (as in Mackert–Lohman's original formulas). The value $\overline{HK}'$ includes an integer number of key values (i.e., $\overline{HK}$) and a fractionary part which represents the $(\overline{HK}+1)$th key occurrences, whose retrieval fills the buffer. This value can be simply estimated as

$$\begin{aligned}\overline{HK}' &\simeq HP^{-1}(B) \\ &= \log_q \left(1 - \frac{B}{NP}\right) = \frac{\log(1 - B/NP)}{\log q}\end{aligned}$$

using the formula (31) also for a noninteger number of retrieved key values.

A further improvement consists in adopting the globally feasible replacement approximation (30) for $HP(HK)$. In this case, the structure of (33) remains formally unchanged but a more appropriate value of $\overline{HK}'$ must be calculated by inverting the feasible approximation (30) as follows:

$$\begin{aligned}\overline{HK}' &= HP^{-1}(B) \\ &= \begin{cases} \log\left(1 - \frac{B}{NP}\right)/\log q & \\ \qquad\qquad \text{if this value is } \leq NK(1-q) & \\ NK\left[1 - \left(1 - \frac{B}{NP}\right)^{\frac{1}{NK(1-q)}}\right] & \\ \qquad\qquad \text{if this value is } \geq NK(1-q) & \end{cases}.\end{aligned} \tag{34}$$

This definition is unambiguous, because it could be verified that the two candidate values of $\overline{HK}'$ are equal or both less or both greater than $NK(1-q) = KP$.

B. Extension of the Model to the Uniform Clustering Case

Let us describe the introduction of the clustering factor in order to generalize the Mackert–Lohman cost model. The validity of formula (33) and of the original one (32) is not *formally* restricted to the case of *total uniformity* (natural clustering). As a matter of fact, the steps followed in its derivation are not contradicted by the usage of a probability q—that a page does not contain a given key value—different from (13) estimated in the presence of *total uniformity*. The only requirement is that q has a constant value per key and per page as ensured, by definition, under the *uniform clustering* assumption which is a generalization of *total uniformity*.

Hence, if KP is constant:

$$q = 1 - \frac{KP}{NK} \tag{35}$$

since the ratio KP/NK represents the constant probability that a randomly chosen key value is present on a randomly chosen page. In terms of the clustering factor CF, q can also be expressed as

$$q = 1 - \frac{TP}{CF \cdot NK} = 1 - \frac{DK}{CF \cdot NP}. \tag{36}$$

Therefore, a straightforward extension of (33) and (34) capable to cover the general case of *uniform clustering* consists in using the value of q (35) or (36) instead of (13).

C. A More Accurate Estimation of the Hit-in-Buffer Probability

We already observed that formulas (32) and (33) are optimistic in assuming replacement of keys in the evaluation of the hit-in-buffer probability. Therefore, we are interested in a more accurate and realistic estimation. The hit-in-buffer probability represents the probability that a page is found in the buffer pool *given that* it *can* be hit by the next key value to be retrieved during the *steady-state*. We assume that HK^* key values (with $\overline{HK} \leq HK^* < HK$) have been retrieved and we are looking for the tuples matching the $(HK^* + 1)$th key value. Considering the events:

- *hit*: a given page *can* be hit by the next key, that is, it contains tuples which *can* match the $(HK^* + 1)$th key value;
- *b*: a given page is found in the buffer pool;

we are thus interested in evaluating the conditioned probability

$$\Pr[b|hit]_{HK^*}. \tag{37}$$

The subscript indicates that HK^* key values have already been retrieved and that the hit-in-buffer probability explicitly depends on that number (i.e., is further conditioned by the fact that HK^* keys have been retrieved). Let *hp* be the event that a given page has been hit by any of the first HK^* key values. We can also consider the events:

- $hp \cap \overline{b}$: a given page has been hit by the first HK^* key values but is not in the buffer;
- $\overline{hp}$: a given page is not yet hit;

Since the pages hit by the first HK^* keys represent a subset of $HP(HK^*)$ out of the NP pages of the relation and the pages in the buffer represent a subset of B out of the $HP(HK^*)$ pages hit, the events *b*, $hp \cap \overline{b}$ and $\overline{hp}$ are mutually exclusive and exhaust all the possible states of a page. Therefore we can apply Bayes' formula [8] to calculate the probability (37):

$$\Pr[b|hit]_{HK^*} = \frac{\Pr[hit|b]\Pr[b]}{\Pr[hit|b]\Pr[b] + \Pr[hit|hp\cap\overline{b}]\Pr[hp\cap\overline{b}] + \Pr[hit|\overline{hp}]\Pr[\overline{hp}]}. \tag{38}$$

The probability that a page can be hit conditioned by its state—which resumes the past history—depends only on the number of key values already hit on the page. In particular, $\Pr[hit|\overline{hp}] = 1$ because the pages not yet hit contain only key values which can all be hit, whereas $\Pr[hit|b] = \Pr[hit|hp\cap\overline{b}] = \Pr[hit|hp]$ because we can assume a constant (average) number, say $HKP(HK^*)$, of key values hit per page hit, due to the LRU replacement policy. Indeed the pages in the buffer can be reasonably considered as a random subset of the pages hit so far and the distribution of the hit key values hit over the pages hit can be considered uniform as well. As a matter of fact, pages which are kept in the buffer for a long time seem to contain more key values hit than the other pages hit on the disk but, when a page fault occurs, the page swap has a re-balancing effect, since in general a page of the buffer with several key values hit is replaced by a page with fewer key values hit (possibly just one) and *vice versa*. Moreover, since $\Pr[hp] = \Pr[b] + \Pr[hp \cap \overline{b}]$, (38) becomes

$$\Pr[b|hit]_{HK^*} = \frac{\Pr[hit|hp]\Pr[b]}{\Pr[hit|hp]\Pr[hp] + \Pr[\overline{hp}]}. \tag{39}$$

Substituting the values

$$\Pr[b] = \frac{B}{NP}$$
$$\Pr[hp] = \frac{HP(HK^*)}{NP}$$
$$\Pr[\overline{hp}] = 1 - \frac{HP(HK^*)}{NP}$$
$$\Pr[hit|hp] = 1 - \frac{HKP(HK^*)}{KP}$$

and simplifying we obtain

$$\Pr[b|hit]_{HK^*} = \frac{B\left(1 - \frac{HKP(HK^*)}{KP}\right)}{NP - HP(HK^*)\frac{HKP(HK^*)}{KP}} \tag{40}$$

where the average number $HKP(HK^*)$ of key values hit per page hit can be computed as

$$HKP(HK^*) = \frac{HK^* \cdot HP(1)}{HP(HK^*)} \tag{41}$$

since the numerator represents the total number of key values hit (on the pages hit), given by the retrieval of HK^* key values.

On the basis of the hit-in-buffer probability evaluated in (40), we can come back to the problem of estimating the number of page fetches required by the query. If

$$HP(1)(1 - \Pr[b|hit]_{HK^*})$$

is the expected value of the fetches required to retrieve all the tuples matching the $(HK^* + 1)$th key value, a correct formula to estimate the I/O cost of the query when $HK > \overline{HK}'$ [as in the second branch of formula (33)] can be evaluated as

$$\begin{aligned} &FP(HK) \\ &= B + \left(\left\lceil \overline{HK}' \right\rceil - \overline{HK}'\right) HP(1)\left(1 - \Pr[b|hit]_{\overline{HK}'}\right) \\ &\quad + \sum_{HK^* = \lceil \overline{HK}' \rceil}^{HK-1} HP(1)(1 - \Pr[b|hit]_{HK^*}). \end{aligned} \tag{42}$$

The first term represents the cost of the *transient* (with whole buffer dimension B considered). The second term approximates the cost of retrieving the remaining duplicates (if any) of the $(\overline{HK}+1)$th key value at the beginning of the *steady-state*. The third term represents the cost due to the retrieval of the remaining key values which completes the query. Formula (42) clearly requires $O(HK)$ steps to evaluate the summation in the third term. In order to obtain a cost model with global complexity $O(1)$ with respect to NP, NK, and HK, we must approximate this summation. Two approximations, called the "mean" and the "stepwise" approximation, are used in this work; they ensure satisfactory accuracy in a wide range of combinations of buffer dimension and data clusterings. The derivations of the two approximations are shown in the Appendix. The two resulting versions of the cost model are the following ones.

"MEAN" APPROXIMATION OF THE MODEL:

$$FP(HK) = \begin{cases} HP(HK) & \text{if } 1 \le HK \le \overline{HK}' \\ & \text{(or } B \ge NP) \\ B + (HK - \overline{HK}')\frac{DK}{CF}\left[1 - \frac{B}{NP}\left(1 - 0.5\frac{CF}{TP}\right)\right] & \text{if } \overline{HK}' < HK \le NK \end{cases}$$

where

$$HP(HK) = NP\left[1 - \left(1 - \frac{\max\{HK, TP/CF\}}{NK}\right)^{\min\{HK, TP/CF\}}\right]$$

$$\overline{HK}' = \begin{cases} \log\left(1 - \frac{B}{NP}\right) / \log\left(1 - \frac{DK}{CF \cdot NP}\right) & \text{if this value is } \le \frac{TP}{CF} \\ NK\left[1 - \left(1 - \frac{B}{NP}\right)^{\frac{CF}{TP}}\right] & \text{if this value is } \ge \frac{TP}{CF} \end{cases} .$$

"STEPWISE" APPROXIMATION OF THE MODEL:

$$FP(HK) = \begin{cases} HP(HK) & \text{if } 1 \le HK \le \overline{HK}' \\ & \text{(or } B \ge NP) \\ B + (HK - \overline{HK}')\frac{DK}{CF}\left(1 - \frac{B}{NP}\frac{NK}{NK - \overline{HK}'} + \frac{\overline{HK}'}{NK - \overline{HK}'}\right) & \text{if } \overline{HK}' < HK \le \overline{HK}'' \\ B + (\overline{HK}'' - \overline{HK}')\frac{DK}{CF}\left(1 - \frac{B}{NP}\frac{NK}{NK - \overline{HK}'} + \frac{\overline{HK}'}{NK - \overline{HK}'}\right) & \\ \quad + (HK - \overline{HK}'')\frac{DK}{CF}\left(1 - \frac{B}{NP}\right) & \text{if } \overline{HK}'' < HK \le NK \end{cases}$$

where $HP(HK)$ and $\overline{HK}'$ are the same as the "mean" approximation and

$$\overline{HK}'' = NK\left[1 - \left(\frac{0.5}{NP}\right)^{\frac{CF}{TP}}\right]$$

is the real number of key values retrieved for which almost all the pages of the relation ($HP(\overline{HK}'') = NP - 0.5$ indeed) have been hit.

TABLE II
Constant Parameters of the Relations Used in Experimental Error Analysis

NT	NP	TP	HK
150000	1000	150	25% NK

It is evident that the "stepwise" approximation of the model is slightly more complex than the "mean" approximation. In the next section, the precision of the two approximations is discussed in comparison with simulation results.

IV. Validation of the Model

In this section we analyze the validity of the two approximations of the model proposed for the prediction of data access cost by an index in a uniformly clustered relation. Analytic estimations provided by the model are compared with the results of retrieval simulations using an LRU buffer. The first subsection concerns the error analysis of our model in the presence of *total uniformity*. Our predictions are also matched against Macket–Lohman's ones. The second subsection applies the error analysis to the more general case of higher clusterings. The third subsection takes into account the effectiveness of the model with very large relations, providing a comprehensive "taste" of its behavior in a real world situation. Errors are computed without taking absolute values: negative values mean underestimation.

A. Error Analysis with Natural Clustering

In this subsection we present simulation results involving the retrieval of 25% of the keys from a thousand-page relation presenting the *natural clustering* induced by uniformity. Table II reports the constant parameters of the relations used for experiments in this subsection and in the next one. Variable parameter values will be introduced in the experiment descriptions.

In the first experiment, index scan cost models are tested against varying buffer dimensions for a relation containing $NK = 1000$ distinct key values. Fig. 3 shows the relative errors provided by the "mean" and the "stepwise" approximations of our model and by Mackert–Lohman's formula (32) plotted versus increasing buffer dimensions (from 0.5% to 100% of the relation). The errors are averaged over a run of five randomly generated queries. The curves corresponding to Mackert–Lohman's model predictions are not marked by any symbol in the figures. The improvement provided by our model with respect to Mackert–Lohman's model is apparent. It is mainly due to the more accurate estimation of the situation at the end of the *transient* (the rounding up of the $\overline{HK}$ value accounts for the saw-toothed course of the error of Mackert–Lohman's model). It can be noticed that the error of the "stepwise" approximation is always very small, while the error of the "mean" approximation increases slightly with increasing buffer dimensions.

The second experiment analyzes the prediction errors for relations with varying key duplication. Fig. 4 shows the results of experiments involving several relation classes, with a number of distinct key values NK ranging from 100 to

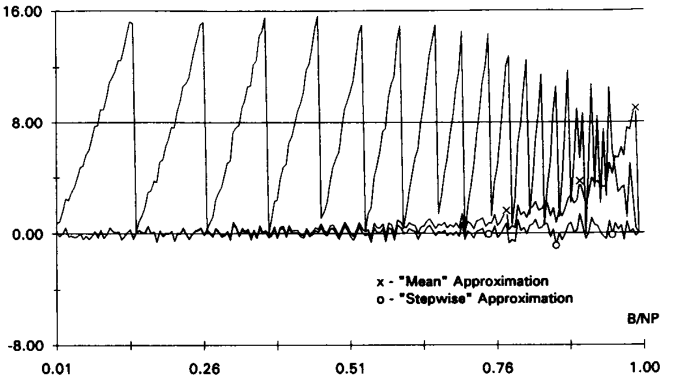


Fig. 3. Percentage errors of the estimations provided by the "mean" and "stepwise" formulas—compared with those of Mackert–Lohman's formula—for varying buffer dimensions. Parameter values are $NK = 1000$, $CF = 1.077$ (*total uniformity*).

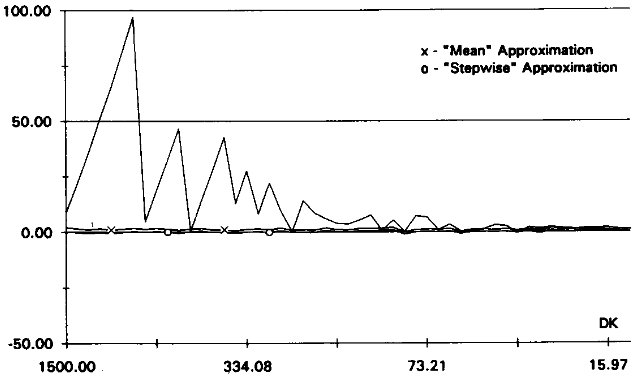


Fig. 4. Percentage errors of the estimations provided by the "mean" and "stepwise" formulas—compared with those of Mackert–Lohman's formula—for varying duplication degrees. Parameter values are $B = 800$, CF as induced by *total uniformity*.

12 000. Two relations per class have been generated and used to run three queries each. The error values plotted in Fig. 4 are averages over the six runs. A buffer of $B = 800$ pages has been used. The behavior of our cost model can be appreciated both in Fig. 4, compared with Mackert–Lohman's, or in Fig. 5, with an enlarged scale. The lack of significance of the Mackert–Lohman prediction formula for high duplications (error near 100%) is due to a poor estimate of the effective *transient* length: as a matter of fact, the plot of the relative error on the transient length estimation between the Mackert-Lohman and our model—$(\overline{HK} - \overline{HK}')/\overline{HK}'$—would present the same shape as the cost error in Fig. 4.

B. Error Analysis with Higher Clustering

In this subsection our attention is focused on the more general case of *uniform clustering*. In this case, no previous reference model exists for comparison. The simulation experiments considered here are similar to those illustrated in the previous subsection, apart from the clustering factor of the relations used, which is higher.

The average relative errors measured in the first experiment (NK fixed to 1000; buffer dimensions ranging from 0.5% to 100% of the relation) are plotted in Fig. 6, whereas the average errors measured in the second experiment (NK ranging from 1000 to 12 000; B fixed to 200) are plotted in Fig. 7. Relative errors do not exceed 6% in both simulations.

The third experiment highlights the error dependence on the clustering factor. It is aimed to fill the gap between the first two experiments of the previous subsection (minimum clustering) and of the present subsection (maximum clustering). For a fixed number of key values $NK = 1000$, relation classes have been generated for increasing clustering factors (CF ranging from 1.25 to 30). Cost models have been tested with three queries on two relations per class and with a buffer of 200 pages. Fig. 8 shows the simulation results, where errors have been averaged over the six runs.

C. Validation for Large Relations and Concluding Remarks

The sample relation R—with schema $R(A_1, A_2, \cdots)$—used for further simulations is characterized by the parameter values reported in Table III. The parameters NK and DK refer to the queried attribute, say A_1, on which the index is built. In particular, the values of A_1 are uniformly distributed over the domain both in the tuples of the relation and in the set of the key values concerned by the queries. High values were chosen for TP and DK in order to obtain very different clustering configurations on the same data.

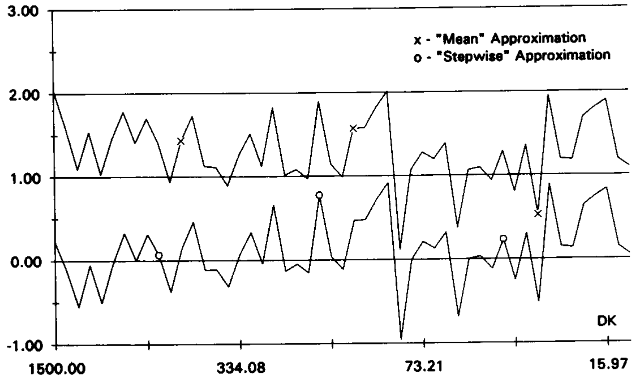


Fig. 5. Percentage errors of the estimations provided by the "mean" and "stepwise" formulas for varying duplication degrees (zoom of the plots in Fig. 4).

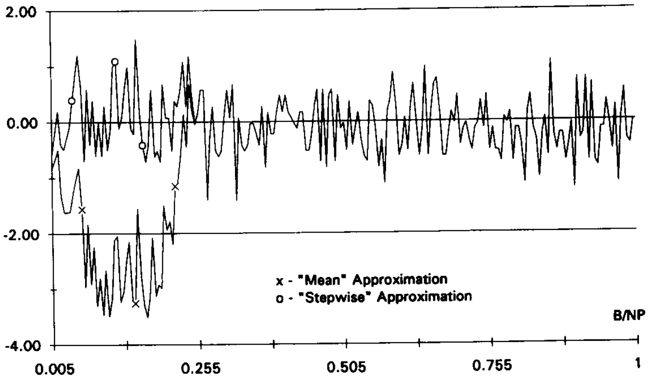


Fig. 6. Percentage errors of the estimations provided by the "mean" and "stepwise" formulas for varying buffer dimensions. Parameter values are $NK = 1000$, $CF = 149.55$ (*total clustering*).

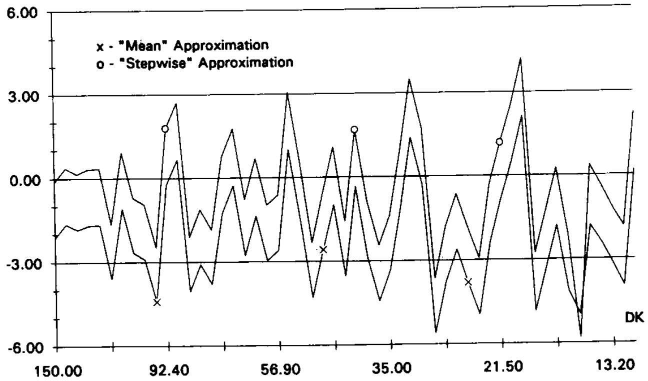


Fig. 7. Percentage errors of the estimations provided by the "mean" and "stepwise" formulas for varying duplication degrees. Parameter values are $B = 200$, CF as induced by *total clustering*.

TABLE III
PARAMETERS OF THE RELATION R. NK AND DK REFER TO THE ATTRIBUTE A_1

NT	NP	TP	NK	DK
1500000	10000	150	10000	150

Two buffer capacities have been used in the simulations: a small buffer of $B = 4000$ pages (equal to 40% of the dimension of R) and a large buffer of $B = 8000$ pages (equal to 80% of the dimension of R).

Three different values of the clustering factor, corresponding to different placements of the tuples on the pages, have been considered, as resumed in Table IV. The lowest clustering corresponds to the case of ***natural clustering induced by uniformity*** of the attribute A_1 in R, which is, for instance, a consequence of the random placement of the tuples on the pages. The highest clustering corresponds to the case of total clustering of the attribute A_1, which occurs, for

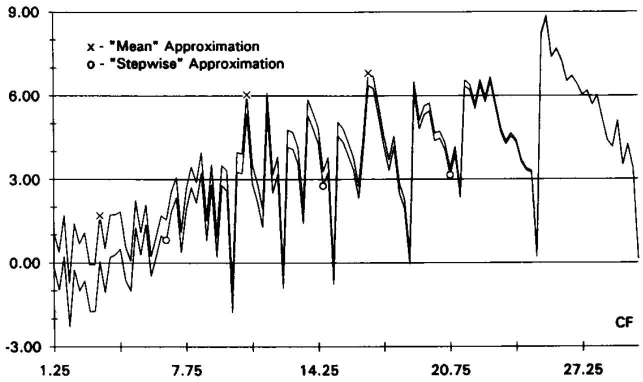


Fig. 8. Percentage errors of the estimations provided by the "mean" and "stepwise" formulas for varying clustering factors. Parameter values are $NK = 1000$, $B = 800$.

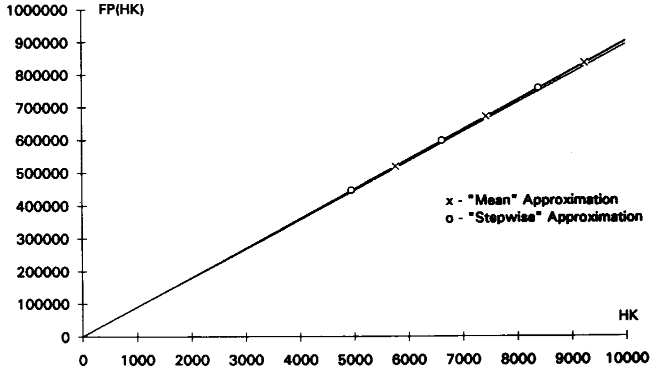


Fig. 9. I/O cost simulations compared with the "mean" and "stepwise" formulas ($B = 4000$ and $CF = 1.01$).

TABLE IV
DIFFERENT CLUSTERINGS OF R CONSIDERED IN SIMULATIONS

Clustering	Placement	KP	CF
Highest	Ordered	1.99	75.25
Medium	Dependent	9.68	15.50
Lowest	Random	149.98	1.01

instance, if the relation R is sorted on the same column. The medium clustering corresponds to an intermediate value of the CF, which can occur in several situations. For instance, we assumed that R is sorted on a second column A_2 and presents the functional dependency $A_1 \longrightarrow A_2$ with *five* different values of A_1 mapped onto a common value of A_2, on average. The combined effect of the ordering on A_2 and of the functional dependency produces the desired clustering of the first attribute. Notice that the three clusterings considered give a representative sample of the possible clusterings which can be found in a relational database. In particular, the ordered placement gives rise to the *theoretical* maximum clustering of an attribute uniformly distributed over the domain, while the random placement gives rise to the minimum.

The application range of our cost model refers both to the cases of indexes used for *set queries* issued on an *ordered attribute* or used for *range* and *set queries* issued on a *unordered* attribute (even if totally clustered). The only case excluded is that of indexed *range queries* on an *ordered attribute*, whose cost can be fairly well approximated by considering a fraction of accessed pages, equal to the fraction of selected keys. This cost is not influenced by the available buffer space since no data page is reused in a single index scan. The curves of the actual cost (not marked by any symbol in figures) are in some cases difficult to distinguish from the predictions of the "mean" and "stepwise" formulas.

The odd-numbered figures from 9 to 19 show the I/O costs due to the number of page fetches $FP(HK)$ needed to retrieve all the tuples matching HK key values, as results from the simulation and from the application of the two approximations of our cost model, for pairs of buffer capacity (4000 and 8000) and clustering degree (lowest, medium, highest).

The even-numbered figures from 10 to 20 display the percentage errors of the previous estimations with respect to the measured I/O cost. Characteristic error values are also summarized in Tables V and VI. Both the figures and the tables show that the percentage error of these trials never reaches 7%. The best approximation is reached sometimes by the "mean," sometimes by the "stepwise" approximation, both proving to be accurate. In any case, the more complex "stepwise" approximation is also shown to be more uniform, ensuring an error always lower than 4% in these experiments.

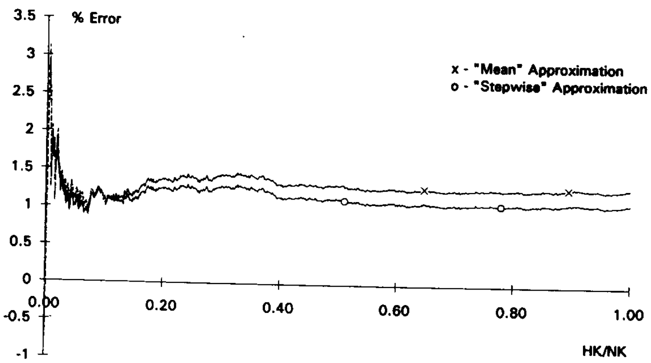


Fig. 10. Percentage errors of the estimations provided by the "mean" and "stepwise" formulas with respect to the actual costs ($B = 4000$ and $CF = 1.01$).

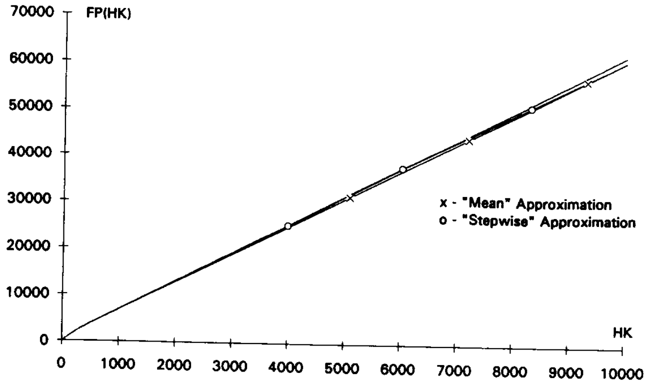


Fig. 11. I/O cost simulations compared with the "mean" and "stepwise" formulas ($B = 4000$ and $CF = 15.50$).

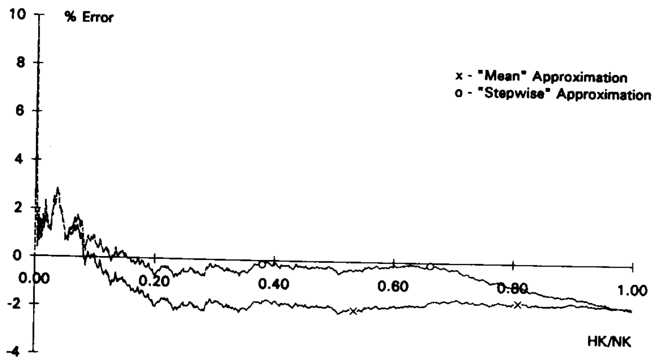


Fig. 12. Percentage errors of the estimations provided by the "mean" and "stepwise" formulas with respect to the actual costs ($B = 4000$ and $CF = 15.50$).

TABLE V
PERCENTAGE ERROR OF THE "MEAN" AND "STEPWISE" APPROXIMATIONS OF THE COST MODEL WITH A SMALL BUFFER ($B = 4000$)

Retrieved Keys	$CF = 1.01$		$CF = 15.50$		$CF = 75.25$	
	"Mean"	"Step."	"Mean"	"Step."	"Mean"	"Step."
25%	1.42	1.28	−1.94	−0.54	−0.21	0.61
50%	1.33	1.14	−1.95	−0.32	−4.51	0.49
75%	1.27	1.07	−1.61	−0.63	−6.06	0.44
100%	1.31	1.11	−1.79	−1.86	−6.75	0.39

TABLE VI
PERCENTAGE ERROR OF THE "MEAN" AND "STEPWISE" APPROXIMATIONS OF THE COST MODEL WITH A LARGE BUFFER ($B = 8000$)

Retrieved Keys	$CF = 1.01$		$CF = 15.50$		$CF = 75.25$	
	"Mean"	"Step."	"Mean"	"Step."	"Mean"	"Step."
25%	−2.29	−3.24	2.39	1.91	0.18	0.18
50%	−2.33	−3.44	1.97	0.87	0.87	0.87
75%	−2.38	−3.56	2.51	−0.67	−0.66	1.28
100%	−2.28	−3.47	2.62	−3.59	−2.27	1.04

A cost model which does not take into account the buffer dimension [e.g., formulas (8) and (25)] always estimates a number of fetches less than or equal to NP, which is inaccurate and becomes meaningless, the lower the clustering.

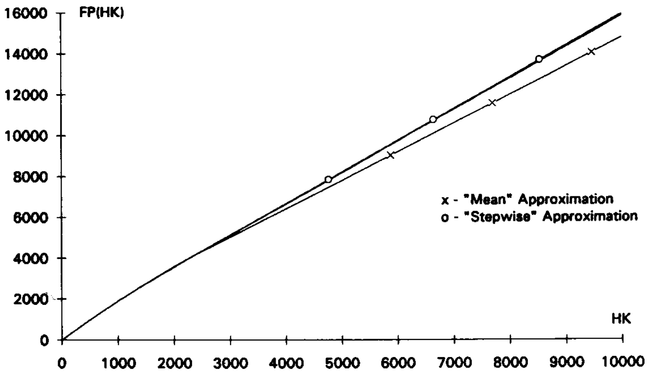


Fig. 13. I/O cost simulations compared with the "mean" and "stepwise" formulas ($B = 4000$ and $CF = 75.25$).

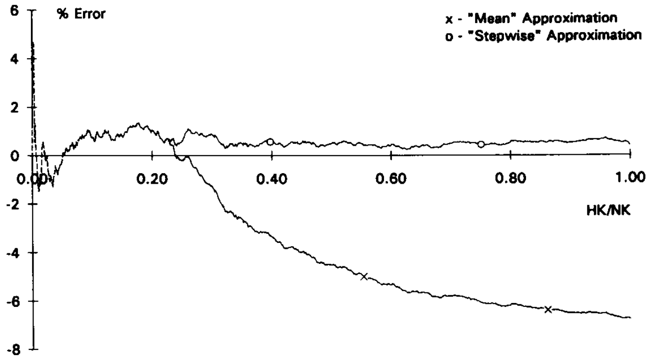


Fig. 14. Percentage errors of the estimations provided by the "mean" and "stepwise" formulas with respect to the actual costs ($B = 4000$ and $CF = 75.25$).

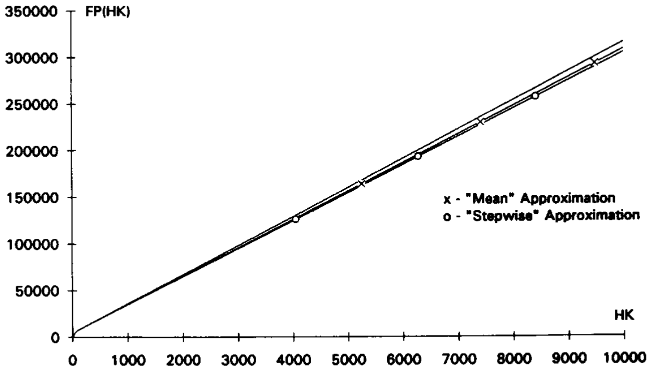


Fig. 15. I/O cost simulations compared with the "mean" and "stepwise" formulas ($B = 8000$ and $CF = 1.01$).

This fact is evident from Figs. 9 and 15, since the actual costs are of higher orders of magnitude with respect to NP. On the other hand, cost models which account for a finite buffer but disregard the actual clustering [e.g., formulas (10), (11), and (14)], in the presence of highly clustered data, give huge overestimations since they implicitly consider *total uniformity*. They produce results similar to those we obtained in Figs. 9 and 15 rather than correct cost estimations as in Figs. 11 and 17 or as in Figs. 13 and 19. Both the approximations of our model are meaningful both in the presence of clustering and when a finite buffer is used; their predictions are also very accurate when the clustering is uniform, as shown by the simulations and the resulting error figures.

Appendix

Derivation of the Approximations Used in the Model

In this Appendix the "mean" and "stepwise" approximations of formula (42), that is the part of the cost model valid when the number of pages referenced by the query exceeds the buffer capacity ($HK > \overline{HK}'$), are presented. The approximations

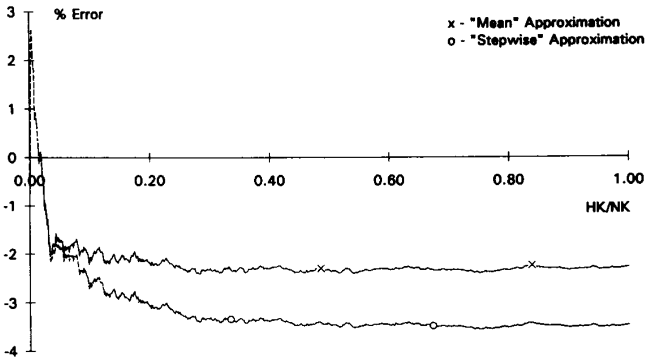


Fig. 16. Percentage errors of the estimations provided by the "mean" and "stepwise" formulas with respect to the actual costs ($B = 8000$ and $CF = 1.01$).

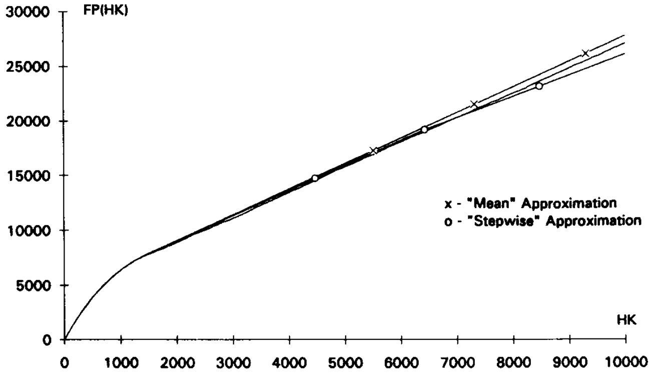


Fig. 17. I/O cost simulations compared with the "mean" and "stepwise" formulas ($B = 8000$ and $CF = 15.50$).

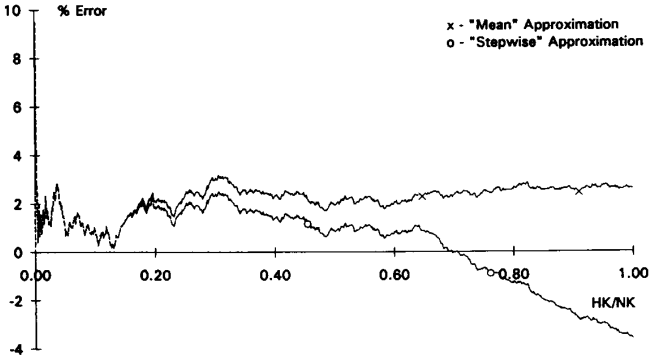


Fig. 18. Percentage errors of the estimations provided by the "mean" and "stepwise" formulas with respect to the actual costs ($B = 8000$ and $CF = 15.50$).

consist in replacing the summation in (42) by one or two terms. Also the second term of formula (42) will be included in the summation replaced. To this end we rewrite formula (42) as follows:

$$FP(HK) = B + \sum_{HK^* = \overline{HK}'}^{HK-1} HP(1)(1 - \Pr[b|hit]_{HK^*}). \quad (43)$$

Notice that such a notation is slightly improper since $\overline{HK}'$, the first value to be assigned to the summation index, is in general noninteger. Nevertheless, this notation highlights the real-valued number of terms which will be replaced.

A. The "mean" Approximation

Our first solution is to replace the summation in (43) with the mean between the first and the last term times the number $(HK - \overline{HK}')$ of terms replaced. Formula (43) becomes

$$\begin{aligned} FP(HK) =& B + (HK - \overline{HK}')HP(1) \\ & \times \left[1 - \frac{\Pr[b|hit]_{\overline{HK}'} + \Pr[b|hit]_{HK-1}}{2}\right]. \quad (44) \end{aligned}$$

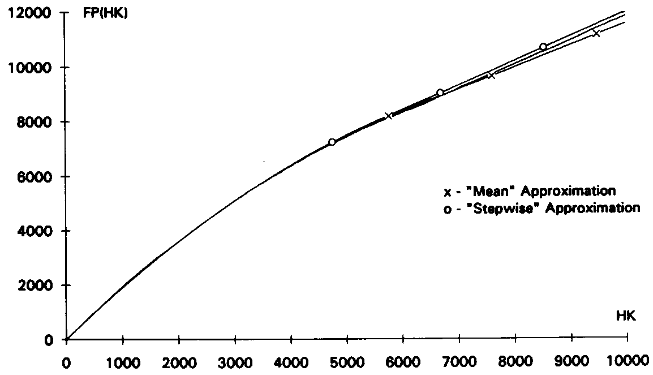


Fig. 19. I/O cost simulations compared with the "mean" and "stepwise" formulas (B = 8000 and CF = 75.25).

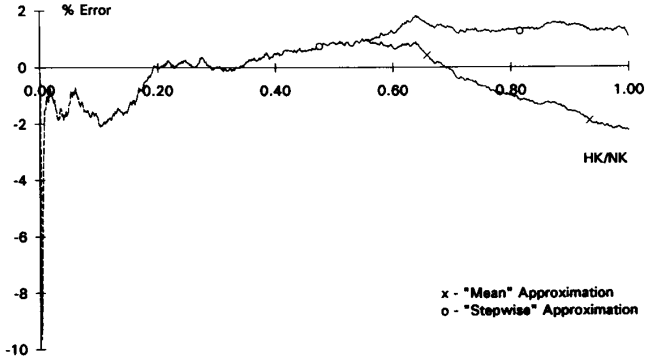


Fig. 20. Percentage errors of the estimations provided by the "mean" and "stepwise" formulas with respect to the actual costs (B = 8000 and CF = 75.25).

Further simplifications are then applied to both probabilities in (44).

The term $\Pr[b|hit]_{\overline{HK}'}$, which represents the hit-in-buffer probability at the end of the *transient*, being $HP(\overline{HK}') = B$, from (40) becomes

$$\Pr[b|hit]_{\overline{HK}'} = \frac{B\left(1-\frac{HKP(\overline{HK}')}{KP}\right)}{NP - B\frac{HKP(\overline{HK}')}{KP}}. \qquad (45)$$

Since $HK > \overline{HK}'$, we know that the buffer is smaller than the relation and therefore $B/NP < 1$. Moreover, the number of key values hit on a buffer page at the end of the transient is smaller than the total number of distinct key values per page[4] and therefore $HKP(\overline{HK}')/KP < 1$. Hence, in general, we can suppose $B/NP \cdot HKP(\overline{HK}')/KP \ll 1$ and neglect the second term in the denominator of (45):

$$\Pr[b|hit]_{\overline{HK}'} \simeq \frac{B}{NP}\left(1 - \frac{HKP(\overline{HK}')}{KP}\right). \qquad (46)$$

If we further suppose one single key value hit per page during the transient, we can simply underestimate $HKP(\overline{HK}')$ as 1 in (46), which yields

$$\Pr[b|hit]_{\overline{HK}'} \simeq \frac{B}{NP}\left(1 - \frac{1}{KP}\right). \qquad (47)$$

The term $\Pr[b|hit]_{HK-1}$ of (44), representing the hit-in-buffer probability just before the last step of the steady-state, when $HK^* = HK - 1$, from (40) is

$$\Pr[b|hit]_{HK-1} = \frac{B\left(1-\frac{HKP(HK-1)}{KP}\right)}{NP - HP(HK-1)\frac{HKP(HK-1)}{KP}}. \qquad (48)$$

If we suppose that during the query execution almost all the pages of the relation have been hit, we have $HP(HK-1) \simeq NP$. This hypothesis becomes valid the longer the query (i.e., referencing a high fraction of key values) and the more widely the key values are spread over the relation (i.e., in the presence of high duplication and low clustering). Thus we can write

$$\Pr[b|hit]_{HK-1} \simeq \frac{B}{NP}. \qquad (49)$$

Finally, by substituting (47) and (49) into (44), we obtain the very simple expression of the "mean" approximation of (42):

$$FP(HK) \simeq B + (HK - \overline{HK}')HP(1)\left[1 - \frac{B}{NP}\left(1 - \frac{0.5}{KP}\right)\right]. \qquad (50)$$

[4] Actually, $HKP(\overline{HK}') \leq KP$ but the only case in which the equality holds is when all the tuples in the buffer have been hit. In general, this occurs when executing a range query over an ordered attribute, but this case has been excluded from the application domain of the model. Otherwise the equality occurrence is quite impossible under the *uniform clustering* assumption.

B. The "stepwise" Approximation

An alternative approximation of (43) consists in dividing the *steady-state* into two periods in which the hit-in-buffer probability is supposed constant. Hence the summation in (43) can be replaced by the sum of the two terms corresponding to these periods. In the first period, which immediately follows the *transient*, we assume a constant hit-in-buffer probability equal to $\Pr[b|hit]_{\overline{HK}'}$, whereas in the latter we assume a constant hit-in-buffer probability equal to B/NP. If we denote with $\overline{HK}''$ the separation point between the two periods, we can write our "stepwise" approximation of (43) as

$$FP(HK) = B + (\overline{HK}'' - \overline{HK}')HP(1)(1 - \Pr[b|hit]_{\overline{HK}'}) + (HK - \overline{HK}'')HP(1)\left(1 - \frac{B}{NP}\right) \quad (51)$$

where $\overline{HK}'$ and $\overline{HK}''$ are noninteger values.

When all the pages of the relation have been hit (at least once), the pages in the buffer contain about the same number of key values hit as the pages virtually returned to the disk. As a matter of fact, we can see from (40) that the hit-in-buffer probability differs from B/NP until all the pages of the relation have been hit. Thus the separation point $\overline{HK}''$ between the two periods can be the real-valued number of key values, for which almost all the pages of the relation have been hit. In particular, we choose the number of key values hit for which the expected number of pages hit (also real) is $NP - \varepsilon$ (e.g., $\varepsilon = 0.5$ has been used in the model). The value of $\overline{HK}''$ can be computed with the approximation *á la* Waters for $HP(HK)$ in the equation:

$$HP(\overline{HK}'') = NP - \varepsilon$$

which yields

$$\overline{HK}'' = NK\left[1 - \left(\frac{\varepsilon}{NP}\right)^{1/KP}\right]. \quad (52)$$

From (41), since $HP(\overline{HK}') = B$, considering (6) and identities $TP = KP \cdot CF$, $DK = NT/NK$, and $TP = NT/NP$ we can write

$$\frac{HKP(\overline{HK}')}{KP} = \frac{\overline{HK}'}{NK}\frac{NP}{B}. \quad (53)$$

Therefore, the term $\Pr[b|hit]_{\overline{HK}'}$ of (51), which can be calculated with (40) and (53), with simple manipulations becomes

$$\Pr[b|hit]_{\overline{HK}'} = \frac{\frac{B}{NP} - \frac{\overline{HK}'}{NK}}{1 - \frac{\overline{HK}'}{NK}} = \frac{B}{NP}\frac{NK}{NK - \overline{HK}'} - \frac{\overline{HK}'}{NK - \overline{HK}'}. \quad (54)$$

Substituting (54) in (51) we obtain the following expression for the "stepwise" approximation of the I/O cost, valid for $HK \geq \overline{HK}''$

$$FP(HK) = B + (\overline{HK}'' - \overline{HK}')HP(1) \times \left(1 - \frac{B}{NP}\frac{NK}{NK - \overline{HK}'} + \frac{\overline{HK}'}{NK - \overline{HK}'}\right) + (HK - \overline{HK}'')HP(1)\left(1 - \frac{B}{NP}\right) \quad (55)$$

in which the value $\overline{HK}''$ of (52) must be used. Clearly, if $\overline{HK}' < HK < \overline{HK}''$, only the first period exists and formula (55) reduces to

$$FP(HK) = B + (HK - \overline{HK}')HP(1) \times \left(1 - \frac{B}{NP}\frac{NK}{NK - \overline{HK}'} + \frac{\overline{HK}'}{NK - \overline{HK}}\right). \quad (56)$$

Introducing in (50), or in (55) combined with (56), the dependence of $HP(1)$ and KP on the clustering factor CF, we obtain the final version of the "mean" and "stepwise" approximations, respectively, of the cost model which have been presented in Section III and tested in experiments in Section IV.

C. An Estimation of the Error Introduced

The probability $\Pr[b|hit]_{HK^*}$, which is expressed in (40), is an increasing function of HK^*, as can be understood from its formulation. From the analytical point of view, if we consider the approximation *á la* Waters of $HP(HK^*)$ given by (25), we can rewrite this probability, after making simple manipulations, as

$$\Pr[b|hit]_{HK^*} = \frac{B}{NP}\frac{1 - \left(1 - \frac{HK^*}{NK}\right)^{KP-1}}{1 - \left(1 - \frac{HK^*}{NK}\right)^{KP}}.$$

Both the numerator and the denominator of the last fraction are increasing functions of HK^*. The numerator is smaller than the denominator but their difference decreases for increasing values of HK^* (the difference vanishes for $HK^* = NK$). Therefore, globally the ratio increases with HK^*. Letting

$$\Pr[b|hit]_0 = \lim_{HK^* \to 0^+} \Pr[b|hit]_{HK^*} = \frac{B}{NP}\left(1 - \frac{1}{KP}\right)$$

$$\Pr[b|hit]_{NK} = \frac{B}{NP}$$

we obtain

$$\Pr[b|hit]_0 < \Pr[b|hit]_{HK^*} \leq \Pr[b|hit]_{NK} \quad (58)$$

for any $HK^* \in (0, NK]$. Notice that the final expression (50) of the "mean" approximation of the cost model could be derived from scratch by assuming a constant hit-in-buffer probability equal to

$$\frac{\Pr[b|hit]_0 + \Pr[b|hit]_{NK}}{2} = \frac{B}{NP}\left(1 - \frac{0.5}{KP}\right), \quad (59)$$

that is, equal to the average between its absolute lower and upper bounds, dependent on the buffer dimension and on the data clustering but regardless of the number of key values HK actually retrieved.

Hence we are able to derive an upper bound for the absolute error introduced by the "mean" approximation:

$$\begin{aligned} E_m &= \left| \sum_{HK^*=\overline{HK}'}^{HK-1} HP(1)(1-\Pr[b|hit]_{HK^*}) \right. \\ &\quad -(HK-\overline{HK}')HP(1) \\ &\quad \left. \times \left[1-\frac{\Pr[b|hit]_0+\Pr[b|hit]_{NK}}{2}\right] \right| \\ &= HP(1) \left| \sum_{HK^*=\overline{HK}'}^{HK-1} \left(\frac{\Pr[b|hit]_0+\Pr[b|hit]_{NK}}{2} \right.\right. \\ &\quad \left.\left. -\Pr[b|hit]_{HK^*}\right) \right| \\ &\le HP(1) \sum_{HK^*=\overline{HK}'}^{HK-1} \left| \frac{\Pr[b|hit]_0+\Pr[b|hit]_{NK}}{2} \right. \\ &\quad \left. -\Pr[b|hit]_{HK^*} \right|. \end{aligned}$$

By means of inequality (58), each term in the summation is smaller than the value[5]

$$\frac{\Pr[b|hit]_{NK}-\Pr[b|hit]_0}{2}=\frac{B}{NP}\frac{0.5}{KP}$$

and thus we can finally write

$$\begin{aligned} E_m &\le HP(1)(HK-\overline{HK}')\frac{B}{NP}\frac{0.5}{KP} \\ &= \frac{1}{2}\left(\frac{HK}{NK}-\frac{\overline{HK}'}{NK}\right)B. \end{aligned} \qquad (60)$$

Therefore, the error E_m will be small if the fraction of key values retrieved in the steady-state is small and, in every case, E_m does not exceed half of the buffer dimension, $B/2$. The error of the "mean" approximation is ensured to be small when the available buffer space is small. On the other hand, when the buffer space is large, the error could be high whereas actual costs are very low. Although a high relative error could be reached in this case, a reliable decision process based on I/O cost comparisons (e.g., access path or index selection) can be correctly supported by the mean approximation of the model: even a poor estimation of the low cost in the presence of a large buffer would never exceed a precise estimation of the cost in the presence of a small buffer, which is higher by orders of magnitude.

As far as the "stepwise" approximation is concerned, we can derive the following upper bound for the absolute error (when $HK > \overline{HK}''$):

$$E_s \le \left(\frac{\overline{HK}''}{NK}-\frac{\overline{HK}'}{NK}\right)B-\Delta_\varepsilon. \qquad (61)$$

The second term Δ_ε corresponds to the error due to the second period of the *steady-state* and can be made negligible by choosing a small value of ε, whereas the first term, which corresponds to the first period, increases when ε decreases since $\overline{HK}''$ approaches NK. Therefore, an appropriate choice of a not too small value of ε can lead to a good tradeoff between the two error components which can compensate each other. The value $\varepsilon = 0.5$ was a reasonable choice in our experiments. The discussion of the dependence of the error on the buffer capacity, which was introduced for the "mean" approximation, applies also to the "stepwise" approximation as well.

However, the experiments described in Section IV clearly demonstrate that the actual errors can be much lower, in realistic situations, than the theoretical upper bounds (60) and (61) just derived.

[5] Note that the error introduced in approximating the hit-in-buffer probability is inversely proportional to KP.

References

[1] M. M. Astrahan, W. Kim, and M. Schkolnick, "Performance of the system *R* access path selection mechanism," in *Inform. Processing 80: Proc. IFIP Congress 80,* Tokyo, Japan, Oct. 1980, pp. 487–491.

[2] S. Bergamaschi and M. R. Scalas, "Choice of the optimal number of blocks for data access by an index," *Inform. Syst.,* vol. 11, no. 3, pp. 199–209, 1986.

[3] F. Bonfatti, D. Maio, M. Spadoni, and P. Tiberio, "An indexing technique for relational data bases," in *Proc. 4th IEEE COMPSAC Int. Comput. Software & Appl. Conf.,* Chicago, IL, Oct. 1980, IEEE, New York, pp. 784–791.

[4] A.F. Cárdenas, "Analysis and performance of inverted database structures," *Commun. ACM,* vol. 18, no. 5, pp. 253–263, May 1975.

[5] S. Christodoulakis, "Estimating block selectivities," *Inform. Syst.,* vol. 9, no. 1, pp. 69–79, 1984.

[6] P. Ciaccia and M. R. Scalas, "Optimization strategies for relational disjunctive queries," *IEEE Trans. Software Eng.,* vol. 15, no. 10, pp. 1217–1235, Oct. 1989.

[7] P. Ciaccia, "Block access estimation for clustered data," *IEEE Trans. Knowledge Data Eng.,* 1993, to be published.

[8] A. B. Clarke and R. L. Disney, *Probability and Random Processes for Engineers and Scientists.* New York: Wiley, 1970.

[9] D. Comer, "The ubiquitous *B*-tree," *ACM Comput. Surveys,* vol. 11, no. 2, pp. 121–137, June 1979.

[10] S. Finkelstein, M. Schkolnick, and P. Tiberio, "Physical database design for relational databases," *ACM Tran. Database Syst.,* vol. 13, no. 1, pp. 91–128, Mar. 1988.

[11] L. F. Mackert and G. M. Lohman, "Index scans using a finite LRU buffer: A validated I/O model," IBM Res. Rep. RJ4836, San Jose, CA, Sept. 1985.

[12] ——, "Index scans using a finite LRU buffer: A validated I/O model," *ACM Trans. Database Syst.,* vol. 14, no. 3, pp. 401–424, Sept. 1989.

[13] D. Maio, M. R. Scalas, and P. Tiberio, "On estimating access costs in relational databases," *Inform. Processing Lett.,* vol. 19, no. 3, pp. 157–161, Oct. 1984.

[14] M. V. Mannino, P. Chu, and T. Sager, "Statistical profile estimation in database systems," *ACM Comput. Surveys,* vol. 20, no. 3, pp. 191–221, Sept. 1988.

[15] P. Palvia and S. T. March, "Approximating block accesses in database organizations," *Inform. Processing Lett.,* vol. 19, no. 2, pp. 75–79, Aug. 1984.

[16] P. Palvia, "The effect of buffer size on pages accessed in random files," *Inform. Syst.,* vol. 13, no. 2, pp. 187–191, 1988.

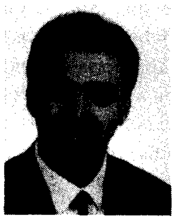
[17] P. G. Selinger, M. M. Astrahan, D. D. Chamberlin, R. A. Lorie, and T. G. Price, "Access path selection in a relational database system," in *Proc. ACM SIGMOD Int. Conf. Management of Data,* Boston, MA, May 1979, ACM, New York, pp. 23–34.

[18] B. T. Vander Zanden, H. M. Taylor, and D. Bitton, "A general framework for computing block accesses," *Inform. Syst.,* vol. 12, no. 2, pp. 177–190, 1986.

[19] K.-Y. Wang, G. Wiederhold, and D. Sagalowicz, "Estimating block accesses in database organizations: A closed noniterative formula," *Commun. ACM,* vol. 26, no. 11, pp. 940–944, Nov. 1983.

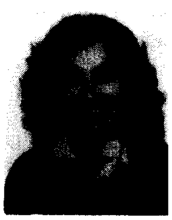
[20] S. J. Waters, "Hit ratios," *Comput. J.,* vol. 19, no. 1, pp. 21–24, 1976.

[21] S. B. Yao, "Approximating block accesses in database organizations," *Commun. ACM,* vol. 20, no. 4, pp. 260–261, Apr. 1977.



Fabio Grandi received the Laurea in electronics engineering from the University of Bologna, Italy, in 1988.

Since 1989 he has worked at the C.I.O.C. center of the Italian National Research Council (CNR), Bologna, Italy, supported by a fellowship from the CNR, in the field of neural networks and temporal databases. He is currently a Ph.D. student at the Department of Electronics, Computer Science and Systems of the University of Bologna. His research interests include databases, storage and access structures and information retrieval systems.



Maria Rita Scalas received the Laurea in physics from the University of Bologna, Italy, in 1974.

From 1975 to 1979 she worked at the Universities of Pisa and Bologna supported by a four-year fellowship from the Italian Ministry of Education. In 1980 she became a Research Associate in Computer Science at the University of Bologna and a consultant at the C.I.O.C.-CNR center of the National Research Council in Bologna. In 1986 she was a visiting scientist at the IBM Scientific Center in Heidelberg, Germany, where she took part in the AIM-P project. In 1987 she became an Associate Professor at the University of Trieste, Italy. She is presently with the Department of Electronics, Computer Science and Systems, University of Bologna. Her scientific interests are in the area of database management systems, temporal databases, access structures, optimizers, and database design.