Minimizing Page Fetches for Permuting Information in Two-Level Storage Part 3. The Paging Model: a Further Improvement on the Floyd Model

TAKAO TSUDA* and KEN-ICHI NAKAGAWA*,**

Part 1 of this paper (this journal, Vol. 6, No. 2, pp. 74–77 (1983)) discussed the generalization of the Floyd model. In this sequel, Part 3, an attempt is made to remove the artificial restrictions inherent in the previously-discussed Floyd model, thereby giving a nontrivial lower-bound to the number of page fetches in a more realistic computational environment. The findings are that the Floyd-model lower bound is reduced by about half, that an algorithm of transposition can in fact be constructed that can do with fewer page fetches than the Floyd-model lower bound and that, hence, the Floyd model is merely an approximation to realistic situations. The Floyd model, inexact as it is, is nevertheless a fairly good approximation of practical importance, because it is easy to analyze and because it incurs only an admissible error of a factor of two at most. In the process of its design, the algorithm of transposition has been found to possess an interesting property of recurrence, whereby the process of transposition can be decomposed to a host of smaller transposition problems. By use of an evidently superior transposition algorithm at the very lowest level, the overall performance of the total transposition algorithm can therefore be significantly improved with regard to the number of page fetches.

1. Introduction

Normally a computer is equipped with a fast main memory and a large but slow auxiliary memory. In cases where data transfers are required between the memories, fast and slow, it is often mandatory that the number of such data transfers be kept as small as possible to expedite the whole process of computation. To date there have been scattered programming techniques to achieve this goal, each being developed *ad hoc* for the type of problem on hand. It may therefore be of great interest if the lower bound for the number of the above data transfers can in principle be computed for a given problem and if this lower bound can actually be realized by programming. The algorithm used in the program will then be the best possible, since there are no other algorithms that surpass the one found with regard to the number of data transfers.

To pursue this vein of approach, the simple Floyd model [1] has recently been generalized so that the model can be applied to actual computational situations [2], [3], [4].*[1] We now review the generalized Floyd model of [3] and expressly indicate the restrictions that there are in it. See Fig. 1. Data in memory space are quantized to pages, each carrying p records (or data). There are p' pages in slow memory, while the main memory can hold w pages ($w \geqq 2$). The computer system under consideration in this paper may or may not be provided with a virtual-memory mechanism; the term 'page' is freely used, meaning the unit of data transfers at a time between the fast and the slow memories. Page transmissions are done as follows.

Step 1. Select w pages in slow memory, and transmit them to main memory.

Step 2. Shuffle the contents of the resident w pages (using a workspace), and then copy them back to the previous w-page locations in slow memory.

Steps 1 and 2 are repeated until the desired permutation of records is established in slow memory. A cycle of Step 1 followed by Step 2 is defined to be w 'operations', which is, in the Floyd model, equivalent to a w-page transfer. Floyd discussed the case where w strictly equals 2. After [2] and [3], let the initial and the final distributions of records in slow memory be denoted by A and B, respectively. The number of operations (or page fetches in the case of the Floyd model) is bounded from below by

$$[V(B|B)-V(A|B)]/p \tag{1}$$

when $w \leqq p$ (see Corollary 1 of [3]) or

$$[V(B|B)-V(A|B)]/e(p) \tag{2}$$

*[1] Reference [2] is a preliminary quick announcement of reference [3]. [2] contains some typographic errors, i.e., all the inequality signs '$>$' in equations (22) and (23) should read '$\geqq$'. All these corrections are entered in [3].

*Department of Information Science, Kyoto University, Kyoto, Japan.

**Now with NEC Corporation.

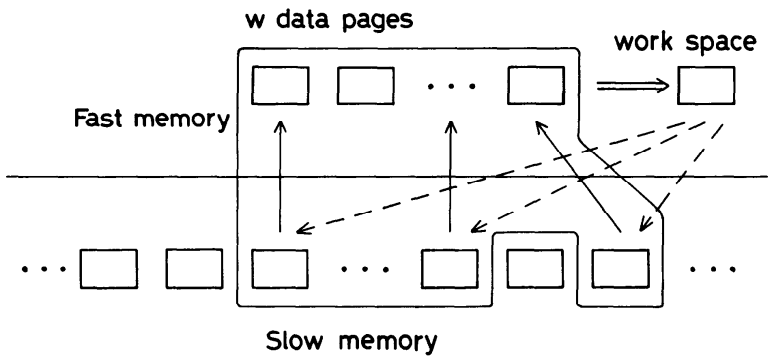


Fig. 1 The generalized Floyd model.

when $w>p$ (see Corollary 4.4 of [2]). Readers are referred to [3] (and [2]) for the definitions and properties of the V- and e-functions appearing in the above.

The purpose of this paper is to remove the artificial restrictions imposed on the Floyd model. As pointed out in Section 2 (p. 184) of [2], page transfers between the two levels of memory take place in lots of w pages, each w pages in slow memory being fetched to main memory, shuffled, and then all pushed together simultaneously to where they had previously been located in slow memory. But rather, out of those w pages fetched to main memory and shuffled at a time, only *one* page could be pushed to slow memory individually and an *entirely new* page be fetched and shuffled with the remaining $(w-1)$ pages. By thus overlapping the resident pages between consecutive passes of shuffling, there will then be more economy in the number of page fetches. Such a page-transfer control reflects the actual situations of what we call demand paging. We are therefore interested in what changes should result in the lower bounds (1) and (2), if such a new model is considered.

2. The Paging Model

2.1 Definition of the Paging Model

The new model described in the preceding section is more realistic in the actual computational environment and will be called the *paging model*. The paging model is defined more precisely as follows.

Step 1. (Initial loading) Fetch w pages in slow memory to main memory.

Step 2. Let the set of the original w pages in slow memory, just duplicated in main memory, be defined as set $\{A\}$. Permute records between the w pages of main memory.

Step 3. Pick one page out of w pages in main memory; then push it back to the location of a page belonging to $\{A\}$.

Step 4. Fetch to main memory one page that is not in $\{A\}$. Keep returning to Step 2 until the objective permutation (i.e., the final distribution of records) is established in slow memory.

In the paging model, a fetch of one page to main memory is considered the unit of data transfer between the two levels of memory.

Remark. Members of set $\{A\}$ vary in time. The previously-defined V-functions of (1) and (2) are state functions defined over the distribution of records in slow-memory pages. Note, however, that the V-functions for the present paging model are state functions that depend on the record distribution in main-memory pages, plus slow-memory pages from which those belonging to set $\{A\}$ are excluded.

2.2 Lower Bounds Based on the Paging Model

Now we will evaluate the lower bounds using the paging model. The lower bound theory of the Floyd model is applied, but, in the following, the e-function (entropy function; e is for entropy.) is defined in a different way, namely,

$$e(x)=x\log_w x, \quad e(0)=0. \tag{3}$$

The e-function thus defined is a continuous function and assumes the same values as the previously defined e-function at those points where x equals a power of w. Recall that the piecewise linear e-function previously defined (see (2) of [3]) is given by

$$e(w^k+l)=kw^k+\left(k+\frac{w}{w-1}\right)l,$$

where

$$k=0, 1, 2, \cdots,$$
$$0\leqq l<(w-1)w^k$$

and

$$e(0)\triangleq 0.$$

The e-function of (3) satisfies the basic relation

$$e\left(\sum_{i=1}^{w} X_i\right) \leqq \sum_{i=1}^{w} \{e(X_i)+X_i\}, \tag{4}$$

where the equality holds only when

$$X_1=X_2=\cdots=X_w=(1/w)\sum_i X_i$$

(see (11) of [3]). Another important property we have seen in (22) of [3], revised for the present case, is given by

$$\begin{aligned} e(X)&>e(X-1)+e(1)\\ &>e(X-2)+e(2)>\cdots\\ \cdots&>e(\lceil X/2\rceil)+e(\lfloor X/2\rfloor), \end{aligned} \tag{5}$$

where a strictly descending order in magnitude excludes the equality signs. Note that (2) of [3] gives, for $w=2$,

$$e(4)+e(7)=e(5)+e(6)=28, \tag{6}$$

whereas (3) gives

$$e(4)+e(7)>e(5)+e(6). \tag{7}$$

The definition of e-function (3) therefore not only renders the function everywhere differentiable, but it removes some ambiguity associated with the process of permuting data between pages. The V-function, giving the total entropy, in (1) and (2) is defined in the same way as in [3]; namely,

$$V(A\,|\,B)=\sum_{i=1}^{p'}\sum_{j=1}^{p'} e(X(i,j)), \tag{8}$$

where $X(i, j)$ is the number of records that are transmitted from the i^{th} page of the initial distribution of records, A, to the j^{th} page of the final distribution of records, $B(i, j=1, 2, \cdots, p')$. Let $\Delta_{\max}$ denote the maximum increment of V-function per page fetch, then the lower bound of total page fetches can be computed by

$$\frac{V(B\,|\,B)-V(A\,|\,B)}{\Delta_{\max}}. \tag{9}$$

2.2.1 The Case of Page Boundaries Ignored in Main Memory

Fig. 2 shows the configurations before and after permuting records in main memory where page boundaries are considered. N indicates the one page *newly* fetched (see Step 4 of the paging model in Section 2.1) whose records are to be permuted with those of $(w-1)$ *old* pages, designated by O, that have been kept resident by the immediately preceding paging. A horizontally stretched rectangle represents a page. Notation $\{i\}$ in each page, where i is an integer either of page numbers $1, 2, \cdots, p'$, stands for the set of those records which should be in page i, the destination page, in the final distribution.

First consider the simple case where the page boundaries of those record pages in main memory are ignored. This simplification is tantamount to the assumption that the cost of permuting data in main memory is negligible. The section that follows treats the case where no such assumption is introduced.

If the page boundaries are ignored in Fig. 2, then the situation will be that shown in Fig. 3. In this case, then, the following relation holds.

Theorem 1. If page boundaries of resident pages are ignored, then the maximum increment of V-function, or the total entropy, per page fetch in the paging model is given by

$$\Delta_{\max}=p\{w-e(w-1)\}. \tag{10}$$

Proof. See Fig. 3. Let the number of records (i.e., size) of set $\{i\}$ of O be denoted by o_i and that of N by n_i. As stated in the proof of Theorem 1 in [3], to increase the entropy maximally it suffices to have those records with a common destination page get together in the same page while forming a new set of pages. In the pres-

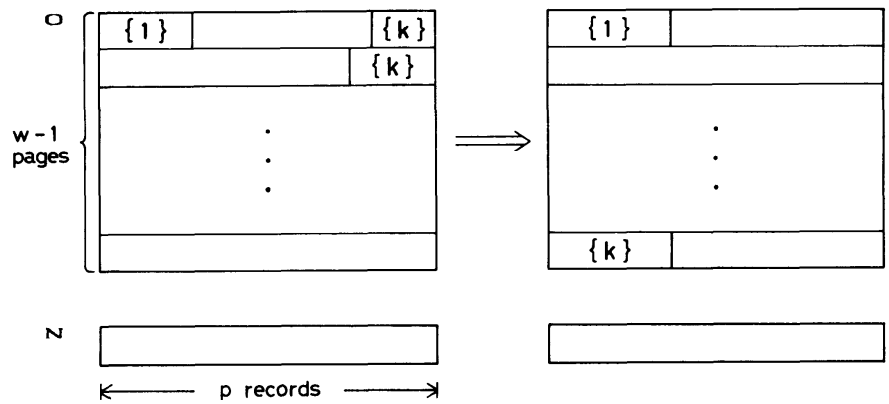
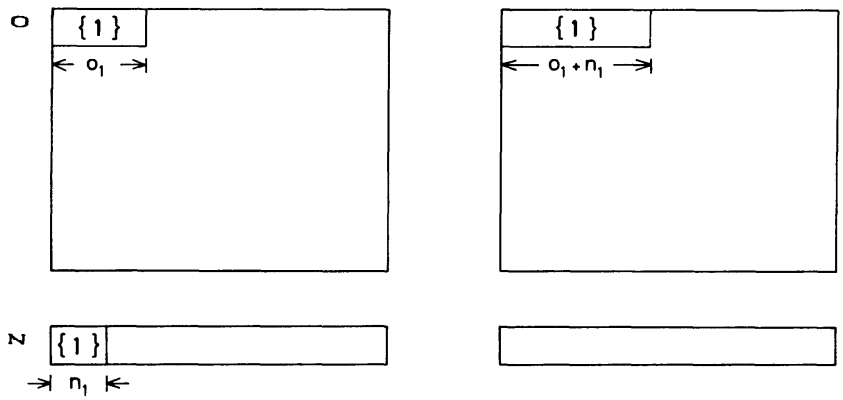


Fig. 2 Configuration of main-memory pages before and after a permutation of records (page boundaries considered).



(a) Before permutation (o_1 and n_1 are the sizes of set {1} in O and N, respectively)

(b) After permutation

Fig. 3 Configuration of main-memory pages before and after a permutation of records (page boundaries ignored).

ent case, therefore, set $\{i\}$ will have size $(o_i+n_i)(\leqq p)$, so that the increment of the relevant e-function is given by

$$e(o_i+n_i)-e(o_i)-e(n_i).$$

If there are m distinct i's ($i \in \{1, 2, \cdots, p'\}$; $2 \leqq m \leqq p'$), then

$$\Delta_{\max}=\underset{2\leqq m\leqq p'}{\operatorname{Max}}\left[\sum_{i=1}^{m}\{e(o_i+n_i)-e(o_i)-e(n_i)\}\right], \tag{11}$$

where there are constraints such that

$$\sum_i o_i=(w-1)p, \tag{12}$$

$$\sum_i n_i=p, \tag{13}$$

$$\left.\begin{aligned} &o_j+n_j\leqq p,\\ &0\leqq o_j,\quad 0\leqq n_j. \end{aligned}\right\} \quad (1\leqq j\leqq m) \tag{14}$$

(11) is a problem of conditional maxima. By solving this, one finds that

$$\Delta_{\max}=p\{w-e(w-1)\}. \tag{15}$$

See Appendix 1 for the derivation. □

2.2.2 The Case of Page Boundaries Considered in Main Memory

In contrast to the assumption made in the preceding section, most computers move data between the two levels of memory by pages, each page retaining its identity before and after transfer. It is therefore necessary to examine the case where page boundaries are considered in main memory. The following holds.

Theorem 2. If page boundaries of resident pages are considered as is usually the case, then the maximum increment of V-function per page fetch in the paging model is given by

$$\Delta_{\max}\begin{cases}=2p & (\text{for } w=2),\\ =p\{e(w-1)-e(w-2)\}+p & (\text{for } w\geqq 3).\end{cases} \tag{16}$$

Proof. (i) *Notations.* In Fig. 4, N indicates the page newly fetched by a paging, while O shows the set of those pages that have remained in main memory. Notations are used such that

$O_i(1)$: the set of those records of page $\#\pi_i$ ($\in \{1, 2, \cdots, p'\}$; $i=1, 2, \cdots, w-1$) that remain in the same page after the current permutation;

$O_i(2)$: the set of those records of page $\#\pi_i$, which move to N after the current permutation;

$N(1)$: the set of those records of N which move to O after the current permutation;

$N(2)$: the set of those records which remain in N after the current permutation.

If $|O_i(1)|$ designates the number of records of set $O_i(1)$ ($i=1, 2, \cdots, w-1$), then

$$|N(1)|=p(w-1)-\sum_i x_i$$

where

$$|O_i(1)| \overset{\Delta}{=} x_i.$$

In the present proof, the incremental variation of the V-function is considered only for the record exchange between the newly-fetched page and the resident $(w-1)$ pages. This is because the shuffling between the remain-

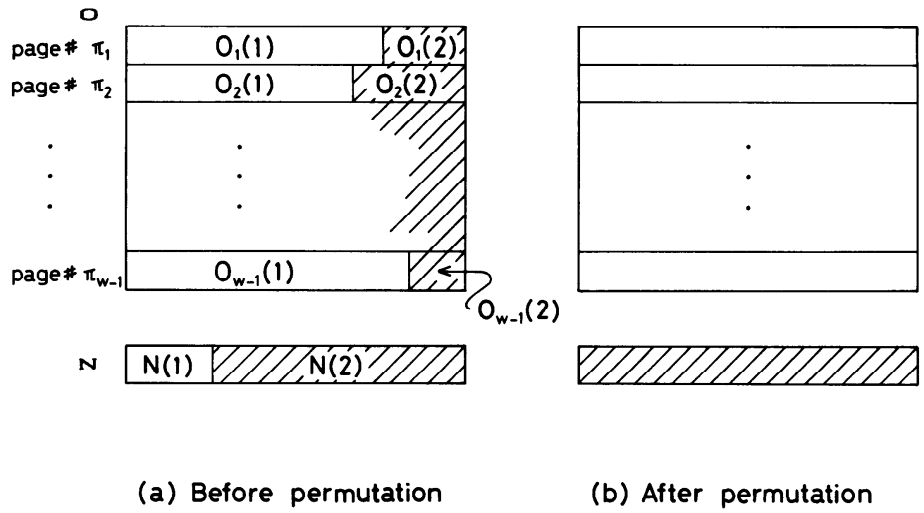


Fig. 4 Main-memory pages before and after a permutation. Page boundaries are not ignored.

ing $(w-1)$ pages can be considered as having been absorbed in those pagings that precede the current one.

(ii) *The increment of the V-function pertaining to the shaded portion of Fig. 4 ($O_i(2)$ and $N(2)$).*

Lemma 1. Let the records of set $O_i(2)$ $(i=1, 2, \cdots, w-1)$ and those of $N(2)$ have $k_i (i=1, 2, \cdots, w-1)$ and k_N distinct destination pages, respectively. Also let the increment of the V-function caused by the record movement of the shaded portion of Fig. 4, i.e., by the movement of those records in sets $O_i(2)$ $(i=1, 2, \cdots, w-1)$ and $N(2)$ be denoted by Δ_2. The increment of entropy, Δ_2, then assumes the greatest value for $k_1=k_2=\cdots=k_{w-1}=k_N$; it follows that

$$\Delta_2 \leqq p. \tag{17}\square$$

For proof of this lemma see Appendix 2.

(iii) *The increment of the V-function pertaining to the blank portion of Fig. 4 ($O_i(1)$ and $N(1)$).* Let x_{ij} and n_{ij} be respectively the number of records of set $O_i(1)$ and that of set $N(1)$ that should reach the destination page $\#j$ $(\in \{1, 2, \cdots, p'\})$ through permuting processes thereafter $(i=1, 2, \cdots, w-1; j=j_1, j_2, \cdots, j_{m_i})$. Here m_i for the i^{th} page of O (i.e., page $\#\pi_i$) is the number of the distinct destination pages of the page's records. Note that $1 \leqq m_i \leqq p$. If the maximum increment of the V-function resulting from the sets of records $O_i(1)$ and $N(1)$ of Fig. 4 is denoted by Δ_1, then finding the value of Δ_1 reduces to the following problem of conditional maxima:

$$\Delta_1 = \operatorname*{Max}_{1 \leqq m_i \leqq p} \left[\sum_{i=1}^{w-1} \sum_{k=1}^{m_i} \{ e(x_{ij_k}+n_{ij_k}) - e(x_{ij_k}) - e(n_{ij_k}) \} \right] \tag{18}$$

on condition that

$$\sum_{k=1}^{m_i} (x_{ij_k}+n_{ij_k}) = p \quad (i=1,2, \cdots, w-1), \tag{19}$$

$$\sum_{i=1}^{w-1} \sum_{k=1}^{m_i} n_{ij_k} \leqq p, \tag{20}$$

$$\left.\begin{aligned} &x_{ij_k}+n_{ij_k} \leqq p, \\ &0 \leqq x_{ij_k}, \\ &0 \leqq n_{ij_k}. \end{aligned}\right\} \begin{aligned} &(i=1,2, \cdots, w-1; \\ &k=1,2, \cdots, m_i) \end{aligned} \tag{21}$$

Leaving the details of demonstration to Appendix 3, one has

$$\Delta_1 \leqq \begin{cases} p & (\text{for } w=2), \\ p\{e(w-1)-e(w-2)\} & (\text{for } w \geqq 3). \end{cases} \tag{22}$$

One thus concludes the proof of Theorem 2, which gives

$$\Delta_{\max} (=\Delta_1+\Delta_2). \qquad \square$$

2.2.3 Estimation of Lower Bounds

There are p' pages of records, each page holding p records, in external slow memory. We want to permute the records with as small a number of page fetches as possible, reordering the initially given distribution of records A to the final and objective distribution B. Since it is tacitly assumed that $p' \gg w$, i.e., main memory size w (in pages) is very small in view of the large amount of data, this process of external rearrangement occurs as a train of partial permutations that will asymptotically approach the final distribution.

Corollary. In the paging model where each page retains its identity in main memory, the lower bound to the number of page fetches required for permuting records from distribution A to B is given by

$$\frac{V(B|B)-V(A|B)}{\Delta_{\max}}+(w-1) \tag{23}$$

where

$$2p \leqq \Delta_{\max} < 2.4p, \tag{24}$$

Table 1 Δ_{max}/p versus w.

w	Δ_{max}/p
2	2.
3	2.26
4	2.38
5	2.397
6	2.396
10	2.36
100	2.21
1000	2.14

the equality sign holding for $w=2$.

Proof. It follows from Theorem 2 that, with w increasing from $w=2$, Δ_{max}/p exhibits the behavior shown in Table 1. It can also be shown that $\Delta_{max}/p \to 2+0$ as $w\to\infty$, so that over all the range of w the value of Δ_{max}/p remains that shown in (24). Formula (9), unlike the Floyd model, does not include the page fetches for the initial loading (cf. Step 1 of Section 2.1). This consideration gives rise to the additional term $(w-1)$ in (23). □

It can therefore be said that, as a rule of thumb, *the paging model reduces the lower bound of the Floyd model by a factor of 2.*

3. A New Algorithm of Transposition

Previously we constructed algorithms of transposition that are the best possible in the sense of the Floyd model [5]. The meaning of the transposition is to reorder $p\times p$ matrix-like data in such a way that data initially stored rowwise in pages are redistributed to those stored columnwise and vice versa. Through the preceding sections of this paper, however, we have learned that there may yet exist some other algorithm that will transcend the Floyd-model lower bound and come closer to that of the paging model of our present interest. We show, in what follows, that we can in fact construct such an algorithm in which substantially fewer page fetches suffice.

Let $f_w(x)$ denote the number of page fetches required for transposing an $x\times x$ matrix where x rows each with x records (or data) are stored in x pages, each page having capacity for holding p records. See Fig. 5 where an example for $x=4$ is given. In the figure, the numbers entered are the elements of relevant X-matrices, not the records (or data) held in each page. It is assumed that at one time the main memory can store w pages ($x\geqq w\geqq 2$) and that page size p is an integral multiple of x. Note that in the X-matrix representation [4] of Fig. 5 the process of diagonalizing the first $x\times x$ matrix can simultaneously diagonalize the other $x\times x$ matrices ($p/x-1$ in number) with no additional page fetches. We then have the following.

Theorem 3. (Recurrence relations of the transposition algorithm). If $f_w(p)$ denotes the number of page fetches required for transposing a $p\times p$ matrix of records, each row stored in a page, with the main memory of w pages in size, and if p is an integral multiple of x and $p>x$

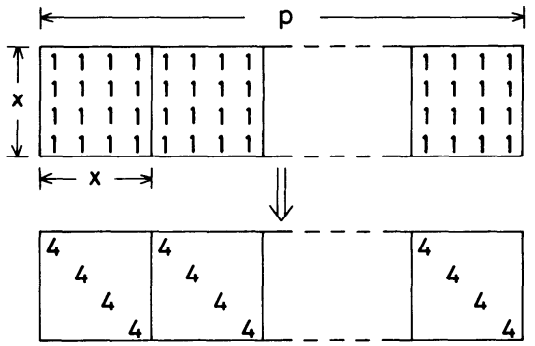


Fig. 5 As the first (left-most) $x\times x$ matrix is diagonalized, the other $x\times x$ matrices are diagonalized simultaneously in main memory. Only non-zero elements are shown.

$\geqq w\geqq 2$, then the following holds.

$$f_w(p)=\frac{p}{x}f_w(x)+xf_w\left(\frac{p}{x}\right). \tag{25}$$

Proof. All the elements of the initial X-matrix have the value of unity (see Fig. 6(a)). This is then transformed to one such as that seen in Fig. 6 (b) with the number of page fetches $(p/x)\times f_w(x)$. This locally diagonalized matrix is reduced to that shown in (c) by renumbering the page numbers in such a way that

page #1 reads page #1,
page #2 reads page #$(p/x+1)$,
page #3 reads page #$(2p/x+1)$,
. . . . . .
page #x reads page #$((x-1)p/x+1)$,
page #$x+1$ reads page #2,
page #$x+2$ reads page #$(p/x+2)$,
page #$x+3$ reads page #$(2p/x+2)$,
. . . . . .
page #$2x$ reads page #$((x-1)p/x+2)$,
page #$2x+1$ reads page #3,
page #$2x+2$ reads page #$(p/x+3)$,
. . . . . .
page #p reads page #p.

Namely, the current page i should read

$$(i-x(\lceil i/x\rceil-1)-1)p/x+\lceil i/x\rceil. \tag{26}$$

As is obvious from Fig. 6 (c), what remains to be done is to transpose a $(p/x\times p/x)$-matrix x times, where each matrix element is an aggregate of x records that move together to the final destination page. (Or each record is viewed as enlarged x times in length.) This process is given by the second term $xf_w(p/x)$ in the right-hand side of (25). We may economize page fetches at least by one page when moving from processes (b) to (c) of Fig. 6 with the overlapping use of some resident pages, but this contribution is very small, hence ignored, in (25). □

Consider next the case where p is a power of w. The Floyd model (p. 76 of [3]) then gives

$$f_w(p)=p\log_w p, \tag{27}$$

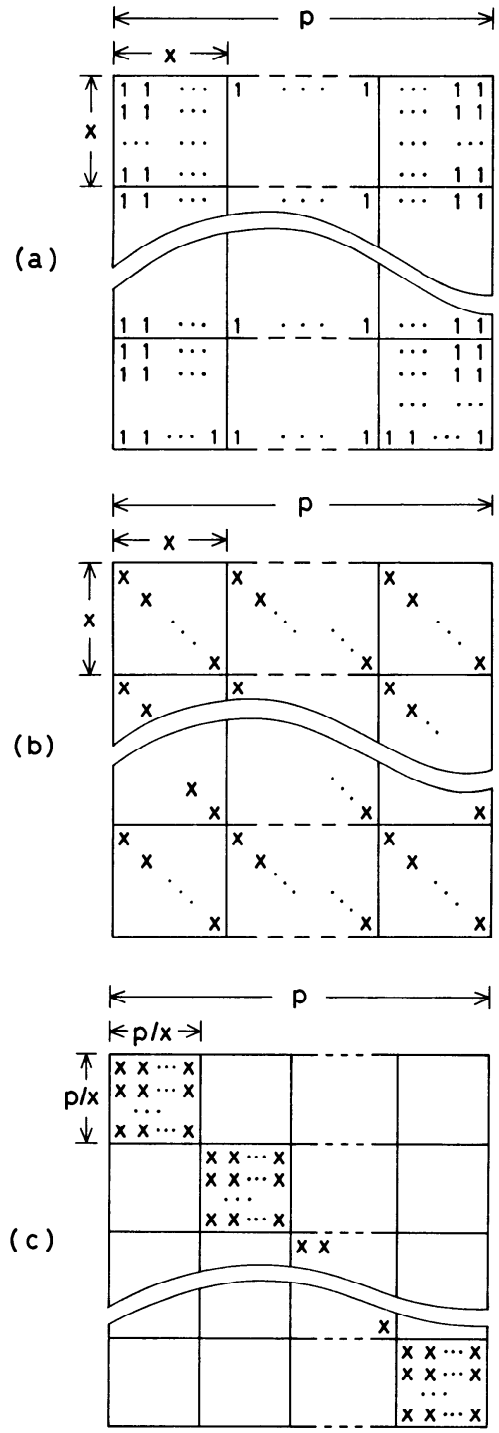


Fig. 6 Visual representation of the recurrence relation of the transposition algorithm using the X-matrix. Only non-zero elements are shown.

and it is interesting that this function satisfies the recurrence of (25). Our strategy to let the paging model have an edge on the Floyd model is to break up the transposition process by the repeated use of recurrence (25), so that the final elementary transpositions of small matrices of records can be done by an algorithm obviously superior to that given by the Floyd model.

By example let $w=2$, and $p=2^4=16$. By use of (25) we have

$$f_2(16)=\frac{16}{4}f_2(4)+4f_2\left(\frac{16}{4}\right)=48. \tag{28}$$

Here note that the paging model gives

$$f_2(4)=6 \quad \text{(paging model)} \tag{29}$$

(see, for this, Fig. 7), whereas the Floyd model gives

$$f_2(4)=8 \quad \text{(Floyd model)}. \tag{30}$$

If the Floyd model is used from the outset (i.e., by not decomposing the whole transposition process like (25)) (see Theorem 1.1 of [5]), then

$$f_2(16)=16\log_2 16=64; \tag{31}$$

or by its slightly improved version

$$f_2(16)=(16-1)\log_2 16+1=61. \tag{32}$$

Compare (28) against (31) and (32). The lower bound given by the Corollary of Section 2.2.3 is, in this case, 34.

If $p=2^5=32$, then

$$\begin{aligned} f_2(32)&=8f_2(4)+4\{2f_2(4)+4f_2(2)\} \\ &=128 \quad \text{(paging model)}, \end{aligned} \tag{33}$$

where obviously $f_2(2)=2$. For this

$$f_2(32)=32\log_2 32=160 \quad \text{(Floyd model)}. \tag{34}$$

If $p=2^6=64$, then

$$\begin{aligned} f_2(64)&=16f_2(4)+4f_2(16) \\ &=288 \quad \text{(paging model)}, \end{aligned} \tag{35}$$

whereas

$$f_2(64)=64\log_2 64=384 \quad \text{(Floyd model)}. \tag{36}$$

Although it is not possible to reach the theoretical lower bound, the paging-model algorithm as combined with decomposition by recurrence gives substantially better results than before.

As an example where p is not a power of w, consider $w=2$, $p=12$. In this case a possible decomposition by recurrence (25) gives,

$$\begin{aligned} f_2(12)&=(12/3)f_2(3)+3f_2(4) \\ &=34 \quad \text{(paging model)}, \end{aligned} \tag{37}$$

where it can be verified in almost the same way as shown in Fig. 7 that

$$f_2(3)=4 \quad \text{(paging model)}. \tag{38}$$

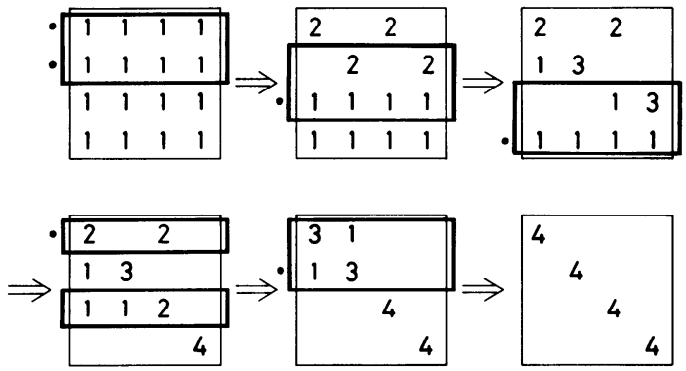


Fig. 7 X-matrix representation of the sequence of transposition f_2 (4) by the paging model. A dot (•) indicates a new page fetched to main memory, while those pages enclosed by horizontally stretched rectangles (drawn by thick lines) are resident pages. Only non-zero elements are shown.

On the other hand the Floyd model algorithm given elsewhere (see Theorem 1.2 of [5]) requires as many page fetches as

$$f_2(12)=56 \qquad \text{(Floyd model)}. \tag{39}$$

4. Concluding Remarks

The paging model introduced in this paper is a more realistic description of data transfers between the fast main memory and the slow but much larger auxiliary memory.

Considering this better model we first assessed what revisions are required of the Floyd model previously studied with regard to the lower bound of the number of page fetches for permuting information between pages in slow memory.

We then derived the recurrence relations of the transposition algorithm. By use of the relations the process of transposition was decomposed into 'low-level' elementary transpositions, and it has been found that the use of the paging-model algorithm at this low level yields page fetches, significantly fewer than the Floyd model, for the whole integrated process of the objective transposition.

One sees from the present study that the lower bound resulting from the generalized Floyd model previously studied is merely an approximation to reality, but, since the Floyd model is easy to use for analysis and its difference from the paging model is quite moderate (a factor of two at most), the generalized Floyd model is still a good approximation of practical importance.

We have referred to the two levels of memory abstractly in this paper as fast and slow. In the system composed of main memory plus magnetic disks the significance of the present study is evident, because data transfers between the two levels are orders of magnitude more time-consuming than CPU operations. Recent supercomputers of parallel pipeline architecture are also equipped with the so-called extended memory made up of fast semiconductor elements. In this situation the term 'page' we have used in this paper can be construed to be the prescribed amount of data transferred at one time to and from the main memory. Even in this case the simple policy of transmitting "the largest amount of data at a time whenever possible" is not adequate, because the ratio of vectorization will most probably be deteriorated by the unoptimized data transfer [6]. We believe that in a few years the lower bound theory of data transfers as studied in the present connotations will be important in large-scale computations by supercomputers.

Appendix 1. Derivation of (15)

Fix m at a certain value. Lagrange's method of indeterminate multipliers is then applied to the contents enclosed by the square brackets of (11). The Lagrangian function F is defined as

$$\begin{aligned} F(\boldsymbol{o}, \boldsymbol{n}, \lambda, \alpha)=&\sum_{i=1}^{m}\{e(o_i+n_i)-e(o_i)-e(n_i)\} \\ &-\lambda_1\left\{\sum_{i=1}^{m} o_i-(w-1)p\right\}-\lambda_2\left\{\sum_{i=1}^{m} n_i-p\right\} \\ &-\sum_{i=1}^{m}\alpha_i(p-o_i-n_i), \end{aligned} \tag{A1}$$

where λ_1 and λ_2 are the Lagrange multipliers and $\alpha_i (i=1, 2, \cdots, m)$ are the generalized Lagrange multipliers with conditions $p-o_i-n_i \geqq 0 (i=1, 2, \cdots, m)$. The solution $(o_i, n_i, \lambda_1, \lambda_2, \alpha_1, \cdots, \alpha_m)$ of the following equations gives the maximum we are looking for.

$$\frac{\partial F}{\partial o_j}=0,$$

$$\frac{\partial F}{\partial n_j}=0,$$

$$\sum_{i=1}^{m} o_i=(w-1)p,$$

$$\sum_{i=1}^{m} n_i=p,$$

$$\alpha_j(p-o_j-n_j)=0,$$

$$\alpha_j\leqq 0,$$

$$o_j+n_j\leqq p.$$

The subscripts j in the above run for $1, 2, \cdots, m$. Definition (3) of $e(x)$ then gives ($o_j\neq 0$, $n_j\neq 0$)

$$\log_w\frac{o_j+n_j}{o_j}=\lambda_1-\alpha_j, \tag{A2}$$

$$\log_w\frac{o_j+n_j}{n_j}=\lambda_2-\alpha_j, \tag{A3}$$

$$\sum_{i=1}^{m} o_i=(w-1)p, \tag{A4}$$

$$\sum_{i=1}^{m} n_i=p, \tag{A5}$$

$$\alpha_j(p-o_j-n_j)=0, \tag{A6}$$

$$\alpha_j\leqq 0, \tag{A7}$$

$$o_j+n_j\leqq p. \tag{A8}$$

It follows from (A2) and (A3) that

$$n_j/o_j=\text{constant}\triangleq\beta \quad (j=1, 2, \cdots, m)$$

where β is independent of j. Using this in (A4) and (A5) we have $\beta=1/(w-1)$, hence

$$o_j=(w-1)n_j. \tag{A9}$$

This, when substituted to (A2) and (A3), gives

$$\left.\begin{aligned}1-\log_w(w-1)&=\lambda_1-\alpha_j,\\ 1&=\lambda_2-\alpha_j\end{aligned}\right\}$$

from which it follows that

$$\alpha_1=\alpha_2=\cdots=\alpha_m\triangleq\alpha \quad \text{(const.)}.$$

We thus see from (A6) there are two cases:

$$\text{(i)}\quad p=o_j+n_j \quad (j=1, 2, \cdots, m) \tag{A10}$$

or

$$\text{(ii)}\quad \alpha=0. \tag{A11}$$

These two cases are examined separately as follows.

(i) $p=o_j+n_j$ $(j=1, 2, \cdots, m)$

Adding (A4) to (A5) we have

$$\sum_{i=1}^{m} o_i+\sum_{i=1}^{m} n_i=wp.$$

Comparing this with (A9), one finds

$$m=w. \tag{A12}$$

On the other hand (A9) and (A10) give

$$\left.\begin{aligned}n_j&=p/w,\\ o_j&=p(w-1)/w\end{aligned}\right\} \tag{A13}$$

at which function F assumes the maximum value. Constraints (12) through (14) define the domain of search to be a bounded closed set in which the greatest maximum exists and is none other than the only maximum just found. Hence, applying (A12) and (A13) to (11), we have

$$\Delta_{\max}=w\left\{e(p)-e\left(\frac{w-1}{w}p\right)-e\left(\frac{p}{w}\right)\right\}. \tag{A14}$$

(ii) $\alpha=0$

In this case the following holds.

$$\begin{aligned}1-\log_w(w-1)&=\lambda_1,\\ 1&=\lambda_2,\\ \Sigma\, n_i&=p,\\ o_j&=(w-1)n_j.\end{aligned}$$

Using these relations we rewrite the contents of the square brackets of (11) as

$$\begin{aligned}&\sum_{i=1}^{m}\{e(o_i+n_i)-e(o_i)-e(n_i)\}\\ &\quad=\sum_{i=1}^{m}\{n_i(w-(w-1)\log_w(w-1))\}\\ &\quad=p\{w-(w-1)\log_w(w-1)\}\\ &\quad=p\{w-e(w-1)\}.\end{aligned} \tag{A15}$$

It is straightforward to verify that (A14) agrees with (A15) by use of (3).

In both cases (i) and (ii) we have thus derived that

$$\Delta_{\max}=p\{w-e(w-1)\}, \tag{A16}$$

and especially for $w=2$

$$\Delta_{\max}=p\{2-e(1)\}=2p. \tag{A17}$$

Appendix 2. Proof of Lemma 1

Suppose

$$\begin{aligned}k_1=k_2=\cdots=k_{j-1}=k_{j+1}=\cdots=k_{w-1}\\ =k_N\triangleq m,\end{aligned} \tag{A18}$$

$$k_j=m+1, \tag{A19}$$

and, moreover, set $O_i(2)$ $(i=1, 2, \cdots, w-1)$ and $N(2)$ have the same common m destination pages, i.e., pages $\#i_1, i_2, \cdots$, and i_m. Extra to these, page $\#i_{m+1}$ is also the destination page of $O_j(2)$. Let $z_{i,1}, \cdots, z_{i,m}$ and $z_{i,m+1}$ denote the number of those records of page $\#\pi_i$ (see Fig. 4) that have the destination pages $\#i_1, \cdots, i_m, i_{m+1}$ respectively. Also $n_1, \cdots, n_m$ indicate that the number of those records of $N(2)$ that are destined to reach page $\#i_1, \cdots, i_m$, respectively. We then have

$$|O_i(2)|=\sum_{r=1}^{m} z_{i,r}=p-x_i \quad (i\neq j), \tag{A20}$$

$$|O_j(2)|=\sum_{r=1}^{m} z_{j,r}+z_{j,m+1}=p-x_j, \tag{A21}$$

$$|N(2)| = \sum_{r=1}^{m} n_r = \sum_{i=1}^{w-1} x_i - (w-2)p. \tag{A22}$$

The entropy variation is then given by

$$\begin{aligned}\Delta_2 &= \left\{e\left(n_1 + \sum_{i=1}^{w-1} z_{i,1}\right) - e(n_1) - \sum_{i=1}^{w-1} e(z_{i,1})\right\} + \cdots \\ &\quad + \left\{e\left(n_m + \sum z_{i,m}\right) - e(n_m) - \sum e(z_{i,m})\right\} \\ &\quad + e(z_{j,m+1}) - e(z_{j,m+1}) \\ &\leqq \sum_{r=1}^{m} n_r + \sum_{r=1}^{m} \sum_{i=1}^{w-1} z_{i,r} \quad \text{(see Eq. (4))} \\ &= \left(\sum_{i=1}^{w-1} x_i - (w-2)p\right) + \sum_{i=1}^{w-1} (p - x_i) - z_{j,m+1} \\ &= p - z_{j,m+1} \leqq p. \quad (\because z_{j,m+1} \geqq 0)\end{aligned} \tag{A23}$$

Conversely, the extra destination page $\#i_{m+1}$ may be one of the destination pages of record set $N(2)$ and the other destination pages $\#i_1, \cdots, i_m$ are common to the m destination pages of $N(2)$ and $O_i(2)$ $(i=1, 2, \cdots, w-1)$. Even in this case we arrive at $\Delta_2 \leqq p$ by similar argument. It is not difficult to prove the lemma in the same way for the general case where the destination page numbers do not completely match between pages of $O_i(2)$ $(i=1, 2, \cdots, w-1)$ and $N(2)$.

Appendix 3. Conditional maximization of (18)

Let m_i be fixed at a certain positive integer. As in Appendix 1, Lagrange's method of indeterminate multipliers is used. The Lagrange function is given by

$$\begin{aligned}F(x, n, \lambda, \alpha) &= \sum_{i=1}^{w-1} \sum_{k=1}^{m_i} \{e(x_{ij_k} + n_{ij_k}) - e(x_{ij_k}) - e(n_{ij_k})\} \\ &\quad - \sum_{i=1}^{w-1} \left\{\lambda_i \left(\sum_{k=1}^{m_i} x_{ij_k} + \sum_{k=1}^{m_i} n_{ij_k} - p\right)\right\} \\ &\quad - \alpha\left(p - \sum_i \sum_k n_{ij_k}\right),\end{aligned} \tag{A24}$$

where λ_i $(i=1, 2, \cdots, w-1)$ are the Lagrange multipliers, and α is the generalized Lagrange multiplier, the latter being for condition (20). Since the domain of search is a bounded closed set given by (21), the maximum, if it is the only maximum, will give the greatest maximum. The following equations are solved for x_{ij_k}, n_{ij_k}, λ_i $(i=1, 2, \cdots, w-1;\ k=1, 2, \cdots, m_i)$ and α.

$$\frac{\partial F}{\partial x_{ij_k}} = 0, \tag{A25}$$

$$\frac{\partial F}{\partial n_{ij_k}} = 0, \tag{A26}$$

$$\frac{\partial F}{\partial \lambda_i} = 0, \tag{A27}$$

$$\alpha\left(p - \sum_i \sum_k n_{ij_k}\right) = 0, \tag{A28}$$

$$p - \sum_i \sum_k n_{ij_k} \geqq 0, \tag{A29}$$

$$\alpha \leqq 0. \tag{A30}$$

With the use of definition (3), the first 3 equations above respectively give the following $(x_{ij_k} \neq 0,\ n_{ij_k} \neq 0)$:

$$\log_w \frac{x_{ij_k} + n_{ij_k}}{x_{ij_k}} = \lambda_i, \tag{A31}$$

$$\log_w \frac{x_{ij_k} + n_{ij_k}}{n_{ij_k}} = \lambda_i - \alpha, \tag{A32}$$

$$\sum_k x_{ij_k} + \sum_k n_{ij_k} = p. \tag{A33}$$

Eliminating λ_i in (A31) and (A32) results in

$$\log_w (n_{ij_k}/x_{ij_k}) = \alpha = \text{constant}$$

or

$$x_{ij_k}/n_{ij_k} = \text{constant} \triangleq \beta. \tag{A34}$$

Substituting x_{ij_k} of (A34) for those of (A31), (A32) and (A33), we then have

$$\log_w \frac{\beta+1}{\beta} = \lambda_i \triangleq \lambda, \tag{A35}$$

$$\log_w(\beta+1) = \lambda - \alpha, \tag{A36}$$

$$\sum_{k=1}^{m_i} n_{ij_k} = p/(\beta+1); \tag{A37}$$

λ and $p/(\beta+1)$ are constants irrespective of i. Next we analyze (A35), (A36), (A37) plus (A28), (A29), (A30). By (A28), either of (i) $\alpha=0$ or (ii) $p = \sum_i \sum_k n_{ij_k}$ holds.

(i) If $\alpha=0$, then (A35) and (A36) give

$$\beta = \pm 1. \tag{A38}$$

If $\beta=-1$, λ is not finite; hence $\beta=1$ gives the solution such that

$$\lambda = \log_w 2. \tag{A39}$$

It then follows from (A37) that

$$\sum_k n_{ij_k} = p/2, \tag{A40}$$

which further gives

$$\sum_i \sum_k n_{ij_k} = \sum_{i=1}^{w-1} \frac{p}{2} = \frac{w-1}{2} p, \tag{A41}$$

a value greater than p insofar as $w>3$. And then this contradicts (A29). The present case (i) therefore holds only for $2 \leqq w \leqq 3$. For w as such, the contents of the square brackets of (18) reduces to

$$\begin{aligned}&\sum_i \sum_k \{e(2n_{ij_k}) - 2e(n_{ij_k})\} \\ &\quad = 2 \log_w 2 \sum_i \sum_k n_{ij_k} \\ &\quad = p(w-1) \log_w 2,\end{aligned} \tag{A42}$$

where (A34) with $\beta=1$ and (A41) are used.

(ii) If $p=\sum_i \sum_k n_{ij_k}$, then this together with (A37) gives

$$\beta=w-2. \tag{A43}$$

w must be such that $w>2$, because $w=2$ gives $\beta=0$, hence $\lambda=-\infty$. Using the relation (A43) in (A37), we have

$$\sum_k n_{ij_k}=p/(w-1). \tag{A44}$$

It thus follows that, for $w>2$, the contents enclosed by the square brackets in (18) results in

$$\sum_i \sum_k \{e((w-1)n_{ij_k})-e((w-2)n_{ij_k})-e(n_{ij_k})\}$$

$$=\sum_i \sum_k \{n_{ij_k}((w-1)\log_w(w-1)-(w-2)\log_w(w-2))\}$$

$$=p\{e(w-1)-e(w-2)\},$$

where again (A34) and (A44) are used.

Combining (i) and (ii) yields the objective solution:

$$\Delta_1 \leqq \begin{cases} p(w-1)\log_w 2 & (\text{for } 2 \leqq w \leqq 3) \\ p\{e(w-1)-e(w-2)\} & (\text{for } w \geqq 3) \end{cases}$$

which is none other than (22).

References

1. FLOYD, R. W. Permuting Information in Idealized Two-Level Storage in *Complexity of Computer Computations* (Miller, R. and Thatcher, J., editors), Plenum Press, New York, 1972, 105–109.
2. TSUDA, T., SATO, T. and TATSUMI, T. Generalization of Floyd's Model on Permuting Information in Idealized Two-Level Storage, *Inf. Process. Lett.* **16**, 4 (1983), 183–188.
3. TSUDA, T., SATO, T. and TATSUMI, T. Minimizing Page Fetches for Permuting Information in Two-Level Storage. Part 1. Generalization of the Floyd Model, *J. Inf. Process.* **6**, 2 (1983), 74–77.
4. TSUDA, T. and NAKAGAWA, K. Minimizing Page Fetches for Permuting Information in Two-Level Storage. Part 2. Design of the Algorithm for Arbitrary Permutations, *J. Inf. Process.* **6**, 2 (1983), 78–86.
5. TSUDA, T. and SATO, T. Transposition of Large Tabular Data Structures with Applications to Physical Database Organization. Part 1. Transposition of Tabular Data Structures, *Acta Inf.* **19** (1983) 13–33.
6. TSUDA, T. and TATSUMI, T. Effective Performance of Vector Processors in the Presence of Memory Hierarchy, *Trans. Inf. Process. Soc. Japan* **25**, 1 (1984) 37–45 (in Japanese).

(Received August 22, 1984; revised January 8, 1985)