MINIMUM NUMBER OF STEPS FOR PERMUTATION IN A BUBBLE MEMORY

by

K. M. Chung
F. Luccio
C. K. Wong

This work was supported in part by the University of Illinois.
MINIMUM NUMBER OF STEPS FOR PERMUTATION IN A BUBBLE MEMORY

K. M. Chung
Wang Laboratories
Lowell, MA 01851

F. Luccio*
Department of Electrical Engineering
and Coordinated Science Laboratory
University of Illinois, Urbana, IL 61801

C. K. Wong
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598

Abstract

We prove a worst case lower bound of \( \frac{n^2}{2} - 5n/2 + 3 \), to the number of steps of any algorithm to permute \( n \) records in the simplest bubble memory structure. By the same argument we prove the optimality of the fastest permutation algorithm known thus far.

Magnetic bubble memories, permutations, optimal algorithm

*On leave from the University of Pisa, 56100 Italy.
1. Introduction

A basic problem in bubble memory systems is the one of generating an arbitrary permutation of records. See for example [2] for a comparative treatment of several permutation algorithms, and for extensive bibliographic references.

Let 0, 1, ..., n-1 indicate the records to be permuted. The simplest memory structure [3] consists of a loop of n positions, also denoted by 0, 1, ..., n-1. The loop is kept in perpetual revolution, such that in a unit of time each record is shifted by one position. At any given time, a permutation $P = p_0, p_1, ..., p_{n-1}$ of 0, 1, ..., n-1 gives the positions of the records in memory, meaning that record $p_i$ is in position $i$, for all $i$.

Two operations $a$ and $b$ are allowed, each requiring a unit of time. The effect of $a$ and $b$ on $P$ is the following (see fig. 1):

\[ a: \quad p_0, p_1, ..., p_{n-1} \rightarrow p_1, ..., p_{n-1}, p_0; \]
\[ b: \quad p_0, p_1, ..., p_{n-1} \rightarrow p_0, p_2, ..., p_{n-1}, p_1. \]

Clearly, $a$ performs a circular shift of all the records, while $b$ performs a circular shift of $p_1, ..., p_{n-1}$, and leaves $p_0$ in its position. To implement operation $b$, a switch is inserted in the loop to "short" the bubble stream between positions 1 and n-1. Operation $a$ (or $b$) is executed by opening (or closing) the switch for a unit of time.

Although this memory system has been studied since several years, none of the permutation algorithms designed for it has been proved to be optimal as yet. The fastest known algorithm, called CLW in the following, was presented in [3]. CLW requires $n^2/2 + O(n)$ steps (i.e., executions of $a$ and/or $b$) in the worst case, while the highest lower bound published thus far is $n^2/8 - O(n)$, which applies to a more powerful memory model [4]. We derive here a lower bound of $n^2/2 - 5n/2 + 3$, thus proving that CLW...
is optimal in the worst case. Moreover, by a similar argument we prove that, in all cases where CLW requires superlinear time, CLW is optimal within a linear additive term.

2. A lower bound to the number of steps

Let \(a^r\) (or \(b^r\)) denote a sequence of \(r\) consecutive applications of operation \(a\) (or \(b\)), and let \(\sigma = a_{r_1}^{s_1}b_{r_2}^{s_2}...a_{r_t}^{s_t}\) be any sequence of operations \((r_1,s_1,...,r_t,s_t \geq 0)\). Also, let \(|\sigma| = \sum_{i=1}^{t} (r_i + s_i)\) be the length of \(\sigma\), and \(P.\sigma\) be the permutation obtained from permutation \(P\) after the application of \(\sigma\).

We always assume that \(0,1,...,n-1\) are the initial positions of records \(0,1,...,n-1\) respectively. That is, \(P_0 = 0,1,...,n-1\) is the initial permutation. For any permutation \(P\) to be constructed, a permutation algorithm must produce a sequence \(\sigma\) such that \(P_0.\sigma = P\). We will derive a lower bound to \(|\sigma|\), in the worst case.

We say that an up occurs in \(P = P_0, P_1, ..., P_n\) for any position \(i\) such that \(p_i > i\), and denote by \(u(P)\) the number of ups in \(P\). (Ups are closely related to permutation rises and Eulerian numbers; see [1,3]). For example, assuming \(n = 8\), the permutation \(A = 2, 1, 3, 7, 0, 4, 5, 6\) has three ups in the positions 0, 2 and 3, that respectively contain the elements 2, 3 and 7.

We now prove a simple basic fact:

**Fact 1** Let \(P.\sigma = Q\), where \(|\sigma| = n-1\). We have: \(u(Q) \leq u(P) + 1\).

**Proof** During the execution of \(\sigma\), each element \(p_i\) of \(P\) falls in one of the following three categories (recall that \(p_i\) occupies position \(i\) in \(P\)):

1. \(p_i\) never occupies position 0 (i.e., when \(p_i\) is in position 1 operation \(b\) occurs in \(\sigma\), taking \(p_i\) to position \(n-1\)). Therefore, \(p_i\) is shifted along a loop of \(n-1\) positions, and appears in \(Q\) in position \(j = i\);
(2) \( p_i \) eventually occupies position 0 (i.e., when \( p_i \) is in position 1 operation \( a \) occurs, taking \( p_i \) to position 0), and is later removed from such a position (i.e., operation \( a \) occurs again before the end of \( \sigma \)). \( p_i \) then appears in \( Q \) in a position \( j > i \);

(3) \( p_i \) occupies position 0 at the end of \( \sigma \), hence in \( Q \).

Any element \( p_i \) of case (1) or (2) occupies a position \( j \geq i \) in \( Q \), hence no such an element can give rise to a new up in \( Q \), if it did not correspond to an up in \( P \). The only element of case (3) can give rise to a new up in \( Q \). Therefore, \( u(Q) \) is at most equal to \( u(P) + 1 \).

Take for example the permutation \( A = 2,1,3,7,0,4,5,6 \) already introduced, and the sequence \( \sigma = ab^{2}a^{3}b \) with \( |\sigma| = 7 = n-1 \). The new permutation \( A.\sigma = 5,2,3,7,1,0,4,6 \) has four ups in positions 0, 1, 2 and 3, that is \( u(A.\sigma) = u(A) + 1 \) (see fig. 2).

The following corollary applies to the initial permutation \( P_0 \):

**Corollary 2.** Let \( P_0.\sigma = Q \), and let \( |\sigma| = h(n-1) \), where \( h \) is a non-negative integer. We have: \( u(Q) \leq h \).

The proof of corollary 2 immediately follows from fact 1, noting that \( u(P_0) = 0 \).

Although the number of ups of a permutation cannot be increased by more than one each \( n-1 \) steps (fact 1), such a number is widely variable during the execution of a sequence \( \sigma \) of arbitrary length. There are permutations, however, whose number of ups is almost constant under any
sequence of applications of operation $a$. In fact we have:

**Fact 3** There exists at least one permutation $S$ such that $u(S \cdot \overline{a}^r) \geq \lfloor (n-1)/2 \rfloor$, for any non-negative integer value $r$.

For example, the permutation $S_1 = n-1, n-2, \ldots, 1, 0$ satisfies fact 2. So do the permutations $S_2 = 0, 2, 4, \ldots, n-2, 1, 3, 5, \ldots, n-1$ (n even) and $S_3 = 0, 2, 4, \ldots, n-1, 1, 3, 5, \ldots, n-2$ (n odd). The simple proof of this claim is left to the reader. We merely note that the inequality in fact 3 could be more strictly rewritten as $u(S \cdot \overline{a}^r) = (n-1)/2$, for $n$ odd; and $u(S \cdot \overline{a}^r) = n/2$, or $u(S \cdot \overline{a}^r) = (n-2)/2$, for $n$ even. This follows from a result of [3]. For example, all the cyclic shifts of the permutation 7, 6, 5, 4, 3, 2, 1, 0 have either four or three ups (see fig. 3).

Any permutation $S$ satisfying fact 3 can now be used to prove a lower bound to the number of steps of any permutation algorithm. In fact, let $S$ be produced from $P \cdot \sigma$ by any algorithm, that is $P \cdot \sigma = S$ for some $\sigma$. Let $\sigma' = \sigma \overline{a}^r$ be the concatenation of $\sigma$ and $\overline{a}^r$, where $r$ is the smallest non-negative integer such that:

$$|\sigma'| = |\sigma| + r = h(n-1),$$

with $h$ non-negative integer.

By corollary 2 we have $h \geq u(P_0, \sigma')$. Since $P_0 \cdot \sigma' = S \cdot \overline{a}^r$, that is $h \geq u(S \cdot \overline{a}^r)$, we have by fact 3:

$$h \geq \lfloor (n-1)/2 \rfloor.$$
From (1) and (2), and by noting that \( r \) is at most \( n-2 \), we conclude that the number of steps \( |\sigma| \) required by any permutation algorithm to produce \( S \) from \( P_0 \) is lower bounded as follows:

\[
|\sigma| = |\sigma'| - r \geq \lfloor (n-1)/2 \rfloor (n-1) - (n-2)
\geq \frac{n^2}{2} - \frac{5n}{2} + 3. (3)
\]

3. Optimality of CLW

Let \( \sigma_{CLW} \) be the sequence of operations performed by the CLW algorithm. It was shown in [3] that the number of steps required by CLW is upper bounded as follows:

\[
|\sigma_{CLW}| \leq \lfloor (n-1)/2 \rfloor (n-1) + 2n - 1. (4)
\]

Comparing bounds (3) and (4), we immediately conclude that, in the worst case, CLW is optimal within an additive term of \( 3n-3 \).

As it may be expected, the same permutations \( S \) satisfying fact 3 that have led up to prove the lower bound (3), give rise to the worst cases for CLW. That is, CLW actually meets the upper bound (4) with equality, when it produces any such a permutation \( S \) from \( P_0 \). However, the argument developed above to derive bound (3) will now be extended to prove that CLW is optimal whenever it requires a superlinear number of steps.

Let \( T \) be a generic permutation produced from \( P_0 \) by a sequence \( \sigma \), that is \( P_0 \cdot \sigma = T \); and let \( r \) be the smallest non-negative integer such that 

\[
|\sigma| + r = h(n-1), \text{ with } h \text{ non-negative integer.}
\]

By corollary 2 we have:

\[
h \geq u(P_0 \cdot a^r) = u(T \cdot a^r), \quad 0 \leq r \leq n-2.
\]

From this we derive:
\[ |\sigma| = h(n-1) - r \geq u(T.a^r)(n-1) - (n-2) \]
\[ \geq \min_{0 \leq t \leq n-1} [u(T.a^t)](n-1) - (n-2), \tag{5} \]

that gives a lower bound to the number of steps of any permutation algorithm to produce T, as a function of the minimum number of ups in all the cyclic shifts of T.

It was proved in [3] that the number of steps required by CLW to produce T satisfies the relation:

\[ |\sigma_{CLW}| \leq \min_{0 \leq t \leq n-1} [u(T.a^t)](n-1) + 2n-1. \tag{6} \]

By comparing relations (5) and (6) we conclude that, for all the permutations T such that \( \min_{0 \leq t \leq n-1} [u(T.a^t)] \) is an increasing function of n, hence the running time \( |\sigma_{CLW}| \) is superlinear, such a time meets the lower bound of (5) within a linear additive term of 3n-3. That is, CLW is optimal in all cases, except possibly for the trivial cases when it runs in linear time.
As an example, assume that \( n = 7 \) and the permutation \( T = 1, 2, 4, 0, 6, 3, 5 \) is to be built from \( P_0 = 0, 1, 2, 3, 6, 5, 6 \). The minimum value of \( u(T.a^t) \) is reached for \( t = 5 \). In fact, \( t.a^5 = 3, 5, 1, 2, 4, 0, 6 \), and \( u(T.a^5) = 2 \) (ups in positions 0 and 1). Algorithm CLW performs the following sequence of operations, to construct \( T.a^5 \) from \( P_0 \), and then to construct \( T \) from \( T.a^5 \) (see [3]):

\[
\begin{align*}
0 & \quad 1 & \quad 2 & \quad 3 & \quad 4 & \quad 5 & \quad 6 & \quad P_0 \\
\overline{b^4} & : & 0 & \quad 5 & \quad 6 & \quad 1 & \quad 2 & \quad 3 & \quad 4 \\
\overline{a} & : & 5 & \quad 6 & \quad 1 & \quad 2 & \quad 3 & \quad 4 & \quad 0 \\
\overline{b} & : & 5 & \quad 1 & \quad 2 & \quad 3 & \quad 4 & \quad 0 & \quad 6 \\
\overline{a}^3 & : & 3 & \quad 4 & \quad 0 & \quad 6 & \quad 5 & \quad 1 & \quad 2 \\
\overline{b}^3 & : & 3 & \quad 5 & \quad 1 & \quad 2 & \quad 4 & \quad 0 & \quad 6 & = T.a^5 \\
\overline{a}^2 & : & 1 & \quad 2 & \quad 4 & \quad 0 & \quad 6 & \quad 3 & \quad 5 & = T \\
\end{align*}
\]

That is: \( \sigma_{CLW} = \overline{b^4} \overline{aba}^3 \overline{b^3} \overline{a}^2 \), and \( |\sigma_{CLW}| = u(T.a^5) (n-1) + (n-5) = 2 \cdot 6 + 2 = 14 \).

The lower bound to the number of steps to produce \( T \) is given by:

\[
u(T.a^5)(n-1) - (n-2) = 2 \cdot 6 - 5 = 7.
\]

Obviously, in this example the linear term greatly influences the gap between \( |\sigma_{CLW}| \) and the lower bound.
References


Fig. 1. The basic operations $a$ and $b$ for $n=8$ (numbers represent positions).

Fig. 2. Deriving $A \sigma$, with $\sigma = a b^2 a^3 b$. 
Fig. 3. Ups in the cyclic shifts of a permutation satisfying fact 3 (ups are underlined).