MATHEMATICS Proceedings A 86 (2), June 20, 1983

Sorting arrays by means of swaps

by N.G. de Bruijn

Eindhoven University of Technology, Dept. of Mathematics & Computing Science, P.O. Box 513, Eindhoven, the Netherlands

Communicated at the meeting of January 31, 1983

ABSTRACT

As a preparation to the study of a particular sorting machine, the sorting of arrays by sequences of swaps is treated in this paper. In particular it is shown that if a sequence of "miniswaps" sorts the worst possible arrangement of an array, then it sorts every arrangement of that array. The case of permutation arrays was treated before by R.W. Floyd.

1. INTRODUCTION

If $(\alpha(1), \ldots, \alpha(n))$ is an array of real numbers, say, and if $1 \leq i < j \leq n$, then the *swap* S_{ij} is the operation that sorts the pair $(\alpha(i), \alpha(j))$. That is, if $\alpha(j) < \alpha(i)$ it interchanges the entries $\alpha(i)$ and $\alpha(j)$, and if $\alpha(i) \leq \alpha(j)$ it leaves them untouched.

We consider a particular kind of sorting programs consisting of sequences of swaps only. Such a sequence is given by a sequence of pairs (i,j), and the sequence is fixed before the actual sorting starts. (One can also think of sorting programs where the selection of the next pair (i,j) depends on what happened during the execution of the previous swaps, or even on comparison of pairs $(\alpha(i), \alpha(j))$ without actually swapping these entries, but these kinds of sorting are not to be considered in this paper.)

We shall make a further restriction: the swaps will be required to be miniswaps, which means that always $j = i + 1$.

A previous paper [2] devoted to such swap sorting programs exclusively dealt with the sorting of arrays where $\alpha(i) \neq \alpha(j)$ for $i \neq j$, in particular permutation mappings. For the applications we now have in mind (see [3]) that restriction is

inadequate. In particular we have to deal with arrays of length $3n$, which contain a minimal value m at least n times and a maximal value M also at least n times. Such a restricted kind of arrays can be sorted by shorter swap sequences than those needed in the general case.

The main result of this paper is Theorem 4.1. It says that if a sequence of miniswaps sorts the anti-sorted arrangement of a given content then it sorts every arrangement of that content (for the notions used in this sentence we refer to section 2). The special case of permutation arrays is due to R.W. Floyd (see [4], section 5.3.4, problem 36, p. 241; solution p. 640; two further proofs are presented in [2]). The proof of Theorem 4.1 will follow the idea of one of the proofs given in [2] for Floyd's theorem.

In section 5 we present a generalization of Floyd's theorem in a different direction. It is restricted to permutation arrays again, but it admits both miniswaps and minicoswaps.

2. SORTING OF ARRAYS

Let W be a linearly ordered set, and let n be a positive integer. Then $M(W,n)$ denotes the set of all mappings of W into $\{1,\dots,n\}$. With $\alpha\in M(W,n)$ we can associate the array $(\alpha(1),\dots,\alpha(N))$.

If $w\in W$, $\alpha\in M(W,n)$ then the number of $i\in\{1,\dots,n\}$ with $\alpha(i)=w$ is called the *frequency* of w in α. If both α and β are in $M(W,n)$ we say that α and β *have the same content* if every $w\in W$ has in α the same frequency as in β.

If n is a positive integer then $P(n)$ will denote the set of all bijective mappings π of $\{1,\dots,N\}$ into itself. The elements of $P(n)$ are called *permutations*. If $\pi\in P(n)$ the array $(\pi(1),\dots,\pi(n))$ is called a permutation array.

If $\alpha\in M(W,n)$, $\pi\in P(n)$ then the composite mapping $\alpha\pi$ is called a *rearrangement* of α. It has the same content as α.

There is at least one $\pi\in P(n)$ such that the composite mapping $\alpha\pi$ is *sorted*, i.e. $\alpha(\pi(i))\le\alpha(\pi(j))$ for all i,j with $1\le i<j\le n$. Since $\alpha\pi$ has the same content as α, the sorted rearrangement $\alpha\pi$ is uniquely determined, in spite of the fact that there may be more than one possibility for π (this happens if at least one $w\in W$ has frequency >1 in α).

If $\alpha\in M(W,n)$ there is also at least one $\pi\in P(n)$ such that $\alpha\pi$ is *anti-sorted*, i.e. $\alpha(\pi(i))\ge\alpha(\pi(j))$ for all i,j with $1\le i<j\le n$.

If $\alpha\in M(W,n)$, and if i,j satisfy $1\le i<j\le n$, we can define a mapping γ as follows: $\gamma(k)=\alpha(k)$ for all k which differ from both i and j,

$$\gamma(i)=\min(\alpha(i),\alpha(j)),\ \gamma(j)=\max(\alpha(i),\alpha(j)).$$

This γ will be denoted by $S_{ij}\alpha$. It is not to be considered as the composition of mappings S_{ij} and α: the S_{ij} is not defined as a mapping $W\to W$. We have to consider S_{ij} as an *operator* that maps $M(W,n)$ into itself.

This operator S_{ij} is called a *swap*. Obviously $S_{ij}\alpha$ has the same content as α.

If $1\le i<n$, $j=i+1$ we call S_{ij} a *miniswap*, and denote it by S_i.

If $1\le i_1<j_1\le n$, $1\le i_N<j_N\le n$ then the composition

$$S_{i_1j_1}\dots S_{i_Nj_N} \tag{2.1}$$

(which is again an operator that maps $M(W,n)$ into $M(W,n)$) is called a *composite swap operator*. The case $N=0$ is included, which means that the identity operator is also considered as a composite swap operator. If all factors in (2.1) are miniswaps, i.e. $j_1=i_1+1,\ldots,j_N=i_N+1$, then (2.1) is called a *composite miniswap operator*.

If T is a composite swap operator and $\alpha\in M(W,n)$, then $T\alpha\in M(W,n)$, and $T\alpha$ has the same content as α.

If $T\alpha$ is sorted then we say that T *sorts* α.

3. A PARTIAL ORDER IN $M(W,n)$

Let W be a linearly ordered set, and let n be a positive integer. If α and β are in $M(W,n)$ we write $\alpha\leq\beta$ if and only if there exists a composite swap operator T such that $\alpha=T\beta$.

This is a partial order relation. Transitivity is trivial, so it suffices to show that if both $\alpha\leq\beta$ and $\beta\leq\alpha$ then $\alpha=\beta$. A way to prove this is to consider the number of inversions. An inversion of α is a pair i,j with $1\leq i<j\leq n$ and $\alpha(j)>\alpha(i)$. It is easy to show that if S is a swap then $S\alpha$ has fewer inversions then α.

We can show that miniswaps preserve this partial order. We first show

THEOREM 3.1. Let $1\leq k<h\leq n$, $1\leq i<n$, $\alpha\in M(W,n)$. Then $S_iS_{kh}\alpha\leq S_i\alpha$.

PROOF. If the sets $\{i,i+1\}$ and $\{k,h\}$ are equal we have $S_iS_{kh}\alpha=S_i\alpha$. If these sets are disjoint then $S_iS_{kh}\alpha=S_{kh}S_i\alpha$ which is $\leq S_i\alpha$ by the definition of $\leq$.

If $S_{kh}\alpha=\alpha$ we have $S_iS_{kh}\alpha=S_i\alpha$; if $S_i\alpha=\alpha$ we use the fact that $S_iS_{kh}\alpha\leq\alpha$ by the definition of $\leq$, and again we get $S_iS_{kh}\alpha\leq S_i\alpha$.

The remaining cases are easily checked by inspection. In each case we see that we get from $S_i\alpha$ to $S_iS_{kh}\alpha$ by one or two swaps. For example, if $i=k$, $i+1<h$, $\alpha(h)\leq\alpha(i+1)\leq\alpha(i)$ then $S_iS_{kh}\alpha$ shows the values $\alpha(h)$, $\alpha(i+1)$, $\alpha(i)$ at i, $i+1$, h, respectively, and $S_i\alpha$ shows the values $\alpha(i+1)$, $\alpha(i)$, $\alpha(h)$. We get from the latter to the former by two swaps, viz. $S_iS_{i+1,h}$.

REMARK. In Theorem 3.1 we cannot replace the miniswap by an arbitrary swap. See [2], Remark 7.2.

THEOREM 3.2. Miniswaps preserve the partial order. That is, if both α and β are in $M(W,n)$, and $\beta\leq\alpha$, $1\leq i<n$ then $S_i\beta\leq S_i\alpha$.

PROOF. Since we can get from β to α by a sequence of swaps it suffices to consider the case where $\beta=S_{kl}\alpha$. Here we can apply Theorem 3.1.

4. COMPOSITE MINISWAP OPERATORS THAT SORT EVERY REARRANGEMENT

Let S be a composite miniswap operator. In order to check that it sorts all arrays of a given content it suffices to check that it sorts the anti-sorted array with that same content. This is shown in the following theorem.

THEOREM 4.1. Let W be a linearly ordered set, and $\alpha \in M(W, n)$. Let α_0 and α_1 be the sorted and the anti-sorted array with the same content as α. Let T be a composite miniswap operator such that $T\alpha_1 = \alpha_0$. Then $T\alpha = \alpha_0$.

PROOF. We can obviously get from α to α_0 by a sequence of swaps. From this it can be derived that we can get from α_1 to α by a sequence of swaps: take β, defined by $\beta(i) = \alpha(n+1-i)$, get from β to β_0 by a sequence of swaps, and this can be reinterpreted as the passage of α_1 to α by a sequence of swaps.

So $\alpha \leq \alpha_1$. If T is a composite miniswap operator we get $T\alpha \leq T\alpha_1$ by repeated application of Theorem 3.2. Since $T\alpha_1 = \alpha_0$ we infer that $T\alpha$ cannot have more inversions than α_0. Since α_0 has none, the theorem follows.

EXAMPLE. We present a case that has something of the flavour of the example to be treated in section 5. We omit the cases $n = 1$ and $n = 2$ which are trivial and exceptional.

Let n be a positive integer, $n > 2$, let q be the largest odd number not exceeding n, and r the largest even number not exceeding n. Define the operators $Q, R, T_1, \ldots, T_n$ by

$$Q = S_2 S_4 S_6 \ldots S_{q-1},\ R = S_1 S_3 S_5 \ldots S_{r-1},$$

$$T_1 = R,\ T_2 = QR,\ T_3 = RQR,\ T_4 = QRQR,\ T_5 = RQRQR, \ldots.$$

We can show that T_n sorts α_1. We shall not present a formal proof of this, but just show what happens in the cases $n = 6$ and $n = 7$. If we observe along which lines the numbers move in figs. 1 and 2, we understand the patterns for general n. The top line in the diagram presents the array we start from, the next line is obtained from this line by application of R, the third one from the second by Q, etc.

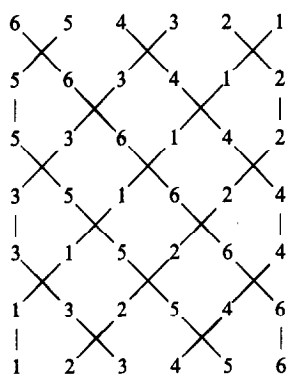

Figure 1

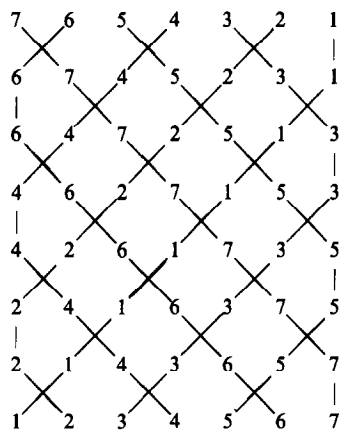

Figure 2

These sorting programs were presented in [4]. (Excercise 37, section 5.3.4; the solution refers to H. Seward (1954), A. Grasselli and to Kautz et al.)

5. A SPECIAL KIND OF ARRAYS

In this section we take a particular kind of arrays that will play a role in the proof (see [3]) that a certain sorting machine invented by Armstrong and Rem (see [1]) sorts the way they claim.

We take $n>1$, and we consider arrays of length $3n$ with values in some linearly ordered set W. W has a minimal element m and a maximal element M, and the elements of the array satisfy

$$(5.1)\qquad \begin{cases} \alpha(1)=\ldots=\alpha(n)=m,\ m\le\alpha(n+1)\le\ldots\le\alpha(2n)\le M, \\ \alpha(2n+1)=\ldots=\alpha(3n)=M. \end{cases}$$

We shall describe a composite miniswap operator that sorts every rearrangement of α. As in section 4, let q be the largest odd number not exceeding n, and r the largest even number not exceeding n. We define

$$Q=S_{2n+2}S_{2n+4}\ \ldots\ S_{2n+q-1},\ R=S_{2n+1}S_{2n+3}\ \ldots\ S_{2n+r-1}$$

(where Q is the identity if $n=2$), and

$$K_i=S_iS_{i+1}\ \ldots\ S_{2n}\quad (i=1,\ldots,2n).$$

With these abbreviations it will turn out that

$$(5.2)\qquad (QK_{2n})R(QK_{2n-1})R\ \ldots\ (QK_2)R(QK_1)$$

sorts every rearrangement of (5.1). But in order to prove a related statement (see (5.4)) we shall economize a bit on (5.2) by deleting some factors that will turn out to be superfluous anyway. The fact that these factors can be deleted corresponds to the triangles of m's and M's in the upper right and lower right corners of fig. 3, where swaps have no effect.

We define, for $m=0,\ldots,4n-2$

$$(5.3)\qquad Z_m=\Pi_{j\in\mathbb{Z}\mid j\ge 0\wedge 0<m-2j<n\wedge j\ge m+1-2n}S_{2n+m-2j}.$$

We agree that (5.3) represents the identity operator when the product is empty (i.e. when $m=0$ or $m=4n-2$). And we note that the factors in the product commute, since generally $S_pS_q=S_qS_p$ if $|p-q|>1$. Therefore we do not have to prescribe the order of the factors in (5.3). We also note that if in (5.3) we would delete the restrictions $j\ge 0$ and $j\ge m+1-2n$ we would have $Z_m=Q$ or R according to m even or odd.

As a refinement of the statement about (5.2) we now claim that

$$(5.4)\qquad (Z_{4n-2}K_{2n})Z_{4n-3}\ \ldots\ (Z_2K_2)Z_1(Z_0K_1)$$

sorts every rearrangement of α (if α satisfies (5.1)). By Theorem 4.1 it suffices to check that it sorts the anti-sorted rearrangement of α. We can prove this by describing how the various entries move in a pattern we depict in fig. 3 for the case $n=4$ with the entries m, m, m, m, a, b, c, d, M, M, M, M where $m\le a\le b\le c\le d\le M$. At the end of each line the operation that produces the next line is shown.

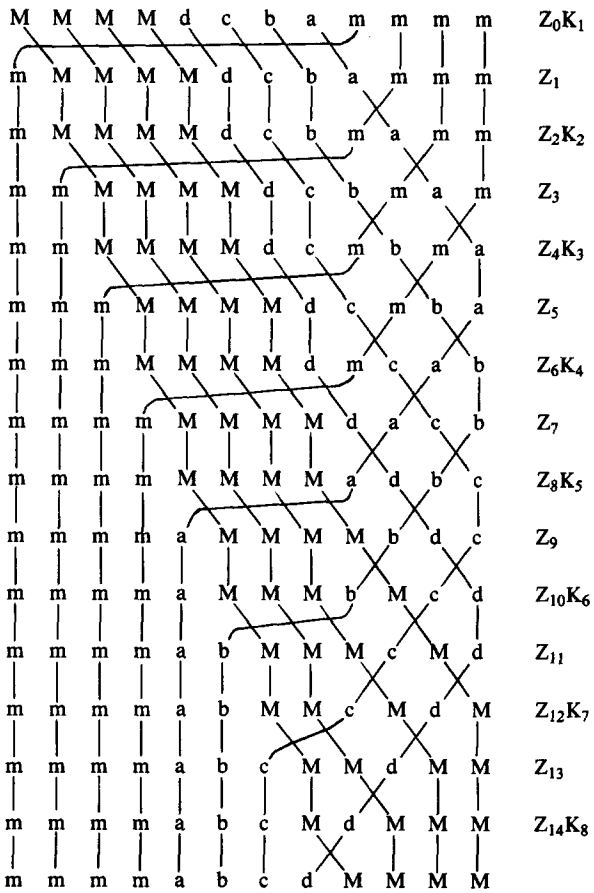

Figure 3

It would not be hard to describe the equivalent of fig. 3 for the case of general n in formal terms. What happens in the last n columns is closely related to figs. 1 and 2.

We shall now use the sorting property of (5.4) in order to show the sorting property of a different expression that we need in a particular application (see [3]). Take the miniswaps

$$S_{2n+1}, S_{2n+2}, \ldots, S_{3n-1}$$

and let V denote the product of these in any arbitrary order (but in all cases where V occurs this is the same order). Then we claim that

(5.5) $\quad K_{2n} V K_{2n-1} \ldots V K_3 V K_2 V K_1$

sorts every rearrangement of α (if α satisfies (5.1)). In order to show this it suffices to prove that (5.5) can be written in the form

(5.6) $\quad TXU$

where X stands for (5.4), and T and U are products of miniswaps. Actually X sorts every rearrangement β of α, so X sorts $U\beta$, so XU sorts β, whence TXU sorts β.

To facilitate the discussion we write (0) instead of any K_i, and (k) instead of S_{2n+k} (if $k>0$). So in (5.5) we have $2n$ (0)'s, $2n-1$ (k)'s (for each $k>0$). In (5.4) there are $2n$ (0)'s, $2n-1$ (1)'s, ..., $n+1$ $(n-1)$'s, and such that between two consecutive (0)'s there is a product of (k)'s with $k>0$. Let us describe the situation for $n=7$. Reading from left to right we have in the intervals between consecutive (0)'s: (1)(2); (1)(3)(2)(4); eight times the group (1)(3)(5)(2)(4)(6); (1)(3)(5)(2)(4); (1)(3)(2); (1).

If $|k-h|>1$ then (k) and (h) commute. Therefore we are not interested in the position of such (k) and (h) with respect to each other. So if it comes to essentials, we describe (5.4) as: $2n$ (0)'s, between each pair of consecutive (0)'s just one (1); between each pair of consecutive (1)'s just one (2), ..., between each pair of consecutive $(n-2)$'s just one $(n-1)$.

We can recognize this structure in (5.5) too. First mark the (1)'s in the V's. To the right of the (1) in the rightmost V there may occur (k)'s with $k>1$. We shift them to the right (they commute with the (0)) and put them into U. Similarly, the (k)'s with $k>1$ occurring to the left of the leftmost (1) are shifted to the left into T. There remain $2n-2$ (2)'s, and again we shift all (k)'s with $k>2$ occurring to the right of this group into U, and those to the left of this group into T. Now there remain $2n-3$ (3)'s between the remaining (2)'s, etc.

Proceeding this way, we finally get (5.5) in the form (5.6), (with an X that represents the same operator as (5.4)), and that guarantees that (5.5) sorts every rearrangement of α.

5. ANOTHER GENERALIZATION OF FLOYD'S THEOREM

As said in section 1, we get Floyd's theorem from Theorem 4.1 if we restrict the arrays to permutation arrays. We shall now show a generalization in a different direction.

First we introduce the word *coswap* (see [2]): If $1\leq p<q\leq n$, $\tau\in P(n)$, then $\omega=C_{pq}\pi$ is obtained from π by sorting the *values* p and q: if a and b are such that $\pi(a)=p$, $\pi(b)=q$, then $\omega(\min(a,b))=p$, $\omega(\max(a,b))=q$, $\omega(k)=\pi(k)$ if $\pi(k)$ is different from both p and q. Such an operation is called a coswap. A *minicoswap* is a coswap C_{pq} with $q=p+1$, and will be abbreviated as C_p.

THEOREM 5.1. If $M_1, \ldots, M_N$ are miniswaps or minicoswaps, (i.e. elements of the set $\{S_1, S_2, \ldots, S_{n-1}, C_1, \ldots, C_{n-1}\}$) and the product $M_1, \ldots, M_N$ sorts the antisorted permutation array, then it sorts every permutation array.

PROOF. In section 7.1 of [2] it was remarked that minicoswaps preserve the partial order. Therefore the proof of theorem 4.1 (which is, in the case of permutation arrays, identical to the first one of the two proofs given for Floyd's theorem in [2]) can still be used for mixed products of miniswaps and minicoswaps.

REFERENCES

1. Armstrong, P.N. and M. Rem – A serial sorting machine. Comput. and Elect. Engng. **9**, 53–58 (1982).
2. Bruijn, N.G. de – Sorting by means of swappings. Discrete Math. **9**, 333–339 (1974).
3. Bruijn, N.G. de – A sorting machine. Proc. Kon. Ned. Akad. v. Wetensch. **A 86** (= Indagationes Mathematicae **45**), 2, 133–137 (1983).
4. Knuth, D.E. – The Art of Computer Programming. Vol. 3: Sorting and Searching. Addison-Wesley, Reading, Mass., 1973.