A NOTE ON THE WORST CASE OF HEAPSORT

by Clyde P. Kruskal and Elia Weixelbaum

Courant Institute of Mathematical Sciences
New York University

February 1981

It is interesting to compare various sorting algorithms based on numbers of comparisons and exchanges. This point is emphasized in Knuth [Kn, sec. 5.3.1]: "... a theoretical study of this subject [counting comparisons] gives us a good deal of useful insight into the nature of sorting processes ..."

The most commonly known O(n log n) comparison-exchange sorting algorithm not needing external storage is heapsort (sometimes referred to as treesort) [Fl], [Wi]. It is relatively easy to calculate the maximum number of exchanges required by heapsort; this note discusses the maximum number of (key) comparisons required.

The version of heapsort considered (Knuth's Algorithm H [Kn]) appears in Appendix A. We obtain in [KW] the worst-case number of comparisons for heapsort, assuming n is one less than a power of two:

$$2n \log(n+1) - 2n - 3 \log(n+1) + 3 \quad \text{if } n \le 7,$$

$$2n \log(n+1) - 2n - 4 \log(n+1) + 6 \quad \text{otherwise.}$$

An algorithm producing input yielding this maximum appears in Appendix B.

APPENDIX A

```
PROCEDURE HEAPSORT (A,N);

BEGIN

  PROCEDURE SIFT (S,BOUND);
  COMMENT this procedure sifts the item in position S to no lower
                                                 than position BOUND;
    BEGIN
      I := S;    J := 2 * I;    X := A[I];
      WHILE J ≤ BOUND DO
        BEGIN
          IF J < BOUND THEN IF A[J] < A[J+1] THEN J := J + 1;
          IF X ≥ A[J] THEN GOTO DONESIFT;
          A[I] := A[J];    I := J    J := 2 * I
        END;
DONESIFT:
      A[I] := X
    END SIFT;


  PROCEDURE CREATE_HEAP;
    FOR P := (N DIV 2) TO 1 STEP -1 DO SIFT (P,N);


  PROCEDURE SELECT;
    FOR K := N TO 2 STEP -1 DO
      BEGIN
        HOLD := A[1];    A[1] := A[K];    A[K] := HOLD;
        SIFT (1,K-1)
      END;


  CREATE_HEAP;
  SELECT
 END
```

APPENDIX B

```
PROCEDURE WORST_CASE_OF_HEAPSORT (A,N);

COMMENT this algorithm produces an array, A, that yields the worst
  case number of comparisons for heapsort, assuming that N is 1
  less than a power of 2.

BEGIN

  PROCEDURE UNSIFT (S,BOUND);
  COMMENT this procedure unsifts the item in position S up to
                                                   position BOUND;
    BEGIN
      I := S;   J := I DIV 2;   X := A[I];
      WHILE J > BOUND DO
        BEGIN  A[I] := A[J];   I := J;   J := I DIV 2  END;
      A[I] := X
    END;

  PROCEDURE REVERSE_SELECT;
    IF N = 1 THEN A[1] := 1
    ELSE IF N = 3 THEN
      BEGIN  A[1] := 3   A[2] := 2   A[3] := 1   END
    ELSE
      BEGIN
        A[1] := 7;   A[2] := 6;   A[3] := 3;   A[4] := 5;
             A[5] := 4;   A[6] := 1;   A[7] := 2;
        FOR L := 4 TO LOG(N+1) DO
          FOR K := 2 ** (L-1) TO (2 ** L) - 2 STEP 2 DO
            BEGIN
              UNSIFT (index of node containing the 1, 1);
              A[1] := K    A[K] := 1
              UNSIFT (index of node containing the 2, 1);
              A[1] := K+1;   A[K+1] := 2
            END
      END

  PROCEDURE CREATE_REVERSE_HEAP;
    FOR P := 1 TO (N DIV 2) DO UNSIFT(index of node containing
                             smallest element in tree rooted by P, P);

  REVERSE_SELECT;
  CREATE_REVERSE_HEAP
END
```

REFERENCES

[Fl] Floyd, R.W., "Treesort 3: Algorithm 245," Communications of the ACM, 7 12(Dec. 1964), 701.

[Kn] Knuth, D.E., The Art of Computer Programming, Vol. 3: Sorting and Searching. Addison-Wesley, Reading, Mass., 1973.

[KW] Kruskal, C.P. and Weixelbaum, E., "A Worst Case Analysis of Heapsort," Technical Report #18, Department of Computer Science, New York University, N.Y., Nov. 1979.

[Wi] Williams, J.W.J., "Heapsort: Algorithm 232," Communications of the ACM, 7, 6(June 1964), 347-348.