This is not easy to derive from the basic Ord-Smith procedure. This is the only occasion the author has had to compare two relational expressions in a Pascal program!

6. PERMUTATIONS WITH REPEATED MARKS

Another related problem is that of generating permutations of marks some of which may be repeated. The procedures of Figure 1 and Figure 3 produce the full $n!$ permutations, with many repetitions. It is usually desirable that these duplicates be eliminated, so that the permutations generated for $p = AAB$ are AAB, ABA and BAA. This is easily accomplished with the new procedure, by what amounts to a backtracking process: at each level we simply skip an element which has already been chosen. It will always be the one chosen immediately prior to the one being considered. A simple condition:

$(i = k)$ **or** $(p[k] <> p[i])$

determines whether the mark in $p[k]$ is a valid choice. The procedure is given in Figure 6.

To adapt the Ord-Smith version also requires similar treatment to that required for backtracking. It requires the rather complex inner loop below.

```
for i := k to n do
  begin
    Swap(p[i], p[k]);
    if (i = k) or (p[k] <> p[i]) then
      begin
        if k = n then ProcessPerm(p, n) else
          Choose(k + 1);
        if i <> n then Reverse(p, k + 1, n)
      end
    else if i = n then Reverse(p, k + 1, n)
  end
```

7. SOME OBSERVATIONS

The title of this paper is somewhat facetious. Of course, this algorithm does reverse the elements of p. However, it does so not with a *Reverse* statement but with a series of distributed *Rotate* statements, and this is its advantage. A call to *Choose* has a null effect on p, and therefore can be skipped if required without any compensating action. When the *Rotates* are combined into a *Reverse*, on the other hand, a call to *Choose* reverses p, so that if the call is to be skipped this reversal must be simulated. This gives complexity to the resulting procedure as the examples above show. The reader is referred to Wilson[7] for further evidence of this.

Acknowledgements

This work was done while the author was on sabbatical leave at Cornell. I should like to thank David Gries, whose sponsored me, and John Hopcroft and Vince Gibbons, who made me so welcome there. Thanks are also due to the referee for his invaluable comments which enabled me to improve the style of the paper.

REFERENCES

1. H. Hitotumatu and K. Noshita, A technique for implementing backtrack algorithms and its applications, *Inform. Process. Lett.*, **8** (4), 174–175 (1979).
2. R. W. Irving, Permutation Backtracking in Lexicographical Order, *Comput. J.*, **27** (4), 373–374 (1984).
3. R. J. Ord-Smith, Generation of permutations in lexicographical order, *Comm. A.C.M.* **11** (2), 117 (1968).
4. R. Sedgewick, Permutation Generation Methods, *Comp. Surveys* **9** (2), 136–164 (1977).
5. J. S. Rohl, Generating permutations by choosing, *Comput. J.* **21** (4), 302–305 (1978).
6. J. S. Rohl, A faster lexicographical N-queens algorithm, *Inform. Process. Lett.* **17** (5), 231–233 (1983).
7. J. M. Wilson, Interrupted permutations in lexicographic order, *Comput. J.* **26** (1), 93 (1983).

Announcements

Submission Deadline 1 OCTOBER 1990

Fourth International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE-91), Waiohai Hotel, Kauai, Hawaii, 2–5 June 1991.

Call for Papers

Sponsored by ACM/SIGART and The University of Tennessee Space Institute, in cooperation with AAAI, IEEE Computer Society, International Association of Knowledge Engineers, Canadian Society for Computional Studies of Artificial Intelligence, International Neural Network Society, and ECCAI.

Submit four copies of an extended abstract written in English, 6–8 pages by 1 October 1990 to Dr. Jim Bezdek, Program Chair IEA/AIE-91, Division of Computer Science, University of West Florida, Pensacola, FL 32514, USA. Phone (904) 474-2784, Fax: (904) 474-2096. e-mail: JBEZDEK @ UWF.BITNET.