AN EFFICIENT VARIATION OF BUBBLE SORT

Wlodzimierz DOBOSIEWICZ
Institute of Informatics, Warsaw University, 00-901 Warszawa, Poland

Received 15 May 1980

Algorithms, sorting

Bubble sort is well known for being one of the worst sorting algorithms. One way to improve its efficiency was described by Batcher [1], however it is so complex and the amount of bookkeeping so huge that this algorithm is not commonly used in single-processor programming.

Bubble sort can be improved in yet another way, which is similar to Shell's version of the insertion sort.

We select a sequence $h_1, ..., h_t$, where $h_t > 1$ and $t = O(\log n)$, n being the number of sorted elements.

In pass i ($i \leqslant t$) the vector to be sorted is traversed from left to right and items distant h_i from each other are compared and exchanged if necessary. After t passes a regular bubble sort should be used to complete the sorting process. This gives the following algorithm:

```
for ℓ := 1 to t do
begin
  inc := $h_\ell$;
  for i := 1 to n − inc do
    if A[i] > A[i + inc]
    then A[i] ↔ A[i + inc] fi
end;
ℓ := n − 1;
while ℓ > 0 do
  begin
   k := 0;
   for i := 1 to ℓ do
     if A[i] > A[i + 1]
     then {A[i] ↔ A[i + 1]; k := i};
   ℓ := k − 1
  end;
```

Obviously, this algorithm works, as the second part of it is just the well-known bubble sort. Its average efficiency is quite good, as illustrated by the Table 1.

The above results were obtained by sorting real numbers generated by a uniform distribution random numbers generator. The tested programs were written in Fortran and run on a CDC Cyber 72. The sequence $h_1 = \lfloor \frac{1}{2}n \rfloor$, $h_{i+1} = \lfloor \frac{3}{4}h_i \rfloor$ was used. The code of Quicksort was copied from [2], while Shellsort was coded after the description given in [3] with increments of form $2^k - 1$. While the version of Quicksort is approximately the best achievable, there is no certainty whether this is true in the case of Shellsort – this algorithm depends on the choice of increments (see [3] for details).

The exact time complexity of the presented algorithm has yet to be determined. The author hopes that this short presentation will encourage some one to work on this problem.

Table 1
Running times of various sorting algorithms (times in ms)

n	Quicksort	Shellsort	Bubblesort	Modified Bubblesort
10	0.86	0.99	0.6	0.58
50	5.84	8.62	14.6	4.62
100	13.38	21.28	57.3	11.87
500	87.81	153.70	1456.6	84.80
1000	199.8	367.0		194.9
2000	441.2	858.2		450.5
10000	2696.8			2743.8

References

[1] D.E. Knuth, The Art of Computer Programming, Vol. 3 (Addison-Wesley, Reading, MA, 1973) 111–114.

[2] R.C. Singleton, An efficient algorithm for sorting with minimum storage, Comm. ACM 12 (3) 185–187.

[3] D.E. Knuth, The Art of Computer Programming, Vol. 3 (Addison-Wesley, Reading, MA, 1973) 86.