the empirical results obtained, namely an $O(N^{\frac{5}{4}})$ average-case bound for Shellsort using any almost geometric increment sequence for which consecutive increments are relatively prime. Any technique used in this proof would probably also work for Sedgewick's increment sequence. Indeed, we conjecture that for $O(\log N)$ increment sequences, a $\Theta(N^k)$ worst-case running time translates into a $\Theta(N^{(k+1)/2})$ average-case running time. This result has been proven for poor increment sequences, where consecutive increments are divisible ($k = 2$), and seems to apply for the increments in Sections 3 ($k = \frac{3}{2}$) and 4($k = \frac{4}{3}$). If this were the case, then combined with the likely (though unproven) lower bound on $\Omega(N^{1+\epsilon/\sqrt{\log N}})$ for the worst-case running time of Shellsort[18] (with $O(\log N)$ increments), we would get an $\Omega(N^{1+\epsilon/\sqrt{\log N}})$ lower bound for the average-case running time of Shellsort.

Another important question concerns the fits themselves. Although they are very accurate in general, for values of N near an increment, the fits are all consistently underestimates, and for values of N halfway between increments, our fits are all consistently overestimates. Indeed, the error in the fit seems to be a fairly smooth curve that oscillates from positive to negative and depends on distance of N from the nearest increment. We have thus far been unable to determine the form of this error.

Finally, and probably most importantly, our results suggest a direct link between the theoretical bounds obtained in Refs 7 and 17 and suggest that some new increment sequence might exist which would make Shellsort run in, say, $O(N^{\frac{9}{8}})$ expected time, and also give a practical improvement. Such an improvement in Shellsort could make it even faster than quicksort.

Acknowledgements

The author would like to thank Paulette Johnson and Sam Shapiro for their assistance in fitting the data. I also thank Bob Sedgewick for comments on an early draft of this paper. This work was supported by an FIU Foundation Summer Research Grant.

M. A. WEISS
School of Computer Science, Florida International University, University Park, Miami, FL 33199, USA

References

1. A. Aho, J. Hopcroft and J. D. Ullman. *Data Structures and Algorithms*. Addison-Wesley, Reading, MA (1983).
2. J. Bentley, Programming pearls. *Communications of the ACM* **30** (9), 754–757 (1987).
3. J. Esakow and T. Weiss, *Data Structures – An Advanced Approach Using C*. Prentice Hall, Englewood Cliffs, NJ (1989).
4. G. Gonnet, *Handbook of Algorithms and Data Structures*. Addison-Wesley, Reading, MA (1984).
5. P. Helman and R. Veroff, *Walls and Mirrors – Intermediate Problem Solving and Data Structures*. Benjamin-Cummings, Menlo Park, CA (1988).
6. E. Horowitz and S. Sahni, *Fundamentals of Data Structures*. Computer Science Press, Rockville, MD (1977).
7. J. Incerpi and R. Sedgewick, Improved upper bounds on Shellson. *Journal of Computer and System Sciences*, **31** (2) 210–224, (1985).
8. D. E. Knuth, *The Art of Computer Programming. Volume 3: Sorting and Searching*. Addison-Wesley, Reading, MA (1973).
9. K. Melhorn, *Data Structures and Algorithms 1: Sorting and Searching*. Springer. New York (1984).
10. V. Pratt, *Shellsort and Sorting Networks*. Garland Publishing, New York (1979). (Originally presented as the author's Ph.D. thesis, Stanford University, 1971).
11. R. Sedgewick, A new upper bound for Shellsort. *Journal of Algorithms* **2** 159–173, (1986).
12. R. Sedgewick, *Algorithms*. Addison-Wesley, Reading, MA (1988).
13. D. L. Shell, A high-speed sorting procedure. *Communications of the ACM* **2** (7) 30–32, (1959).
14. B. Singh and T. L. Naps, *Introduction to Data Structures*. West Publishing Co, St Paul, MN (1985).
15. H. F. Smith, *Data Structures – Form and Function*. Harcourt Brace Jovanovich, New York (1987).
16. D. Stubbs and N. Webre, *Data Structures with Abstract Data Types and Pascal*. Brooks/Cole, Monterey, CA (1989).
17. M. A. Weiss and R. Sedgewick, More on Shellsort Increment Sequences, Information Processing Letters 34 267–270 (1990).
18. M. A. Weiss and R. Sedgewick, Journal of Algorithms 11 242–251 (1990).
19. *UNIX Programmers Manual*.

Announcements

8–12 April 1991

Fourth International Joint Conference on the Theory and Practice of Software Development, Brighton

TAPSOFT '91 – the Fourth International Joint Conference on the Theory and Practice of Software Development – will be held in Brighton, UK, 8–12 April 1991.

TAPSOFT '91 will include colloquia on 'Trees in Algebra and Programming (CAAP)' and on 'Combining Paradigms for Software Development', a number of keynote talks from invited speakers and a programme of tutorials on topics of interest to those attending the advanced seminars.

Colloquium on Trees in Algebra and Programming (CAAP)

The following topics will be included:

- **Logical, algebraic and combinatorial properties of discrete structures** (strings, trees, graphs, etc.) including the theory of **formal languages**, considered in the broad sense as that of sets of discrete structures, and the theory of **rewriting systems** over these objects.
- **Application** of discrete structures in Computer Science: **syntax and semantics** of programming languages, **operational semantics, logic programming, algorithms and data structures, structures, complexity** of algorithms and **implementation** aspects, **proof techniques** for non-numerical algorithms, **formal specifications, visualisation** of trees and graphs, etc.

Colloquium on Combining Paradigms for Software Development

A major feature of research in software engineering over the past few years has been the trend towards unification and synthesis, combining theory and practice and merging hitherto diverse approaches.

Examples include:

- **Types, objects and databases**, using ideas and techniques from type checking and type inference as developed in programming language semantics.
- **Abstract interpretation** and other **semantics-based techniques** for the compile-time analysis of programs, combining theoretical work on semantics with practical issues in language implementation.
- **Specification of systems** from multiple points of view, perhaps **combining different formalisms,** to provide a more effective basis for software development.
- **Applying process analysis and description** techniques to study software development itself as a formal object of enquiry.

For further information contact:

TAPSOFT '91 Secretariat, PPL Conference Services, 2 Savoy Hill, London WC2R 0BP, UK. Tel: 071-240-1871, ext. 222. Telex: 261176 IEELDN G. Fax: 071-497 3633.