Table 2. A comparison of the actual running times (measured in seconds of CPU time) on a Sun-4/280 computer

n	Er's	%	Semba's	%	Setpart1	%	Setpart2	%
12	3.4	100	19.4	606	7.6	238	7.0	219
13	21.7	100	124.9	576	48.4	223	44.5	205
14	146.1	100	845.4	579	323.7	222	297.0	203
15	1030.4	100	6031.4	585	2285.1	222	2113.8	205
16	7577.1	100	45065.8	595	16825.8	222	15627.6	206

partitions). The results show clearly that both setpart1 and setpart2 are faster than Er's on the VAX computer (94 and 80%, respectively; a similar result is obtained on a Sun-3/180 computer). But they are much slower (more than twice) than Er's recursive one on the Sun-4 computer. The reason is that the recursive call and return consume more time than arithmetic operation on the VAX computer while on the Sun-4 computer, where an RISC (reduced instruction set computer) architecture is adopted, very fast call/return operations (comparable to register arithmetic operation) are provided for a small program like setpart by the aid of a good optimising compiler and fast registers (cf. Ref. 5). Actually, the recursive program compiled under the optimiser option is about four times faster than one under a no-optimizer option, while iterative programs are run twice as fast through compilation under the optimiser option (thus both programs run in a comparable time in a no-optimiser option). We note that this outstanding performance of Sun-4 could be rapidly reduced when the nesting of recursive calls goes deeper than a certain critical depth.[8]

4. Concluding remarks

We have succeeded in deriving an efficient algorithm for generating set partitions. The algorithm is significantly faster than the previous fastest iterative algorithm; it is also faster than the previously reported fastest program, that of Er, on some computers. It is as simple as the programs given by Er[2] and Semba.[6] Though it runs much more slowly than Er's one on some computers (where recursive calls are optimised under an RISC architecture), the present algorithm has one more advantage over the recursive algorithm:[2] it enables an efficient adaptive and cost-optimal parallel algorithm to be devised, as described in another paper by the same authors.[1]

Acknowledgement

We appreciate the comments made by Dr Takio Kurita on the experimental data.

B. DJOKIĆ,[1] M. MIYAKAWA,[2]*
S. SEKIGUCHI,[2] I. SEMBA[3] and
I. STOJMENOVIĆ[4]

[1] Department of Mathematics and Computer Science, University of Miami, P.O. Box 249085, Coral Gables, FL 33124, USA.
[2] Electrotechnical Laboratory, 1-1-4 Umezono Tsukuba 305, Japan.
[3] Ibaraki University, 2-1-1 Bunkyou, Mitoshi, Ibaraki 310, Japan.
[4] Department of Computer Science, University of Ottawa, 34 G. Glinksi, Ottawa, Canada K1N 6N5.
* To whom correspondence should be addressed

References

1. B. Djokić, M. Miyakawa, I. Semba, S. Sekiguchi and I. Stojmenović, Parallel algorithms for generating subsets and set partitions to appear in *The Computer Journal*.
2. M. C. Er, A fast algorithm for generating set partitions. *The Computer Journal*, **31** (3), 283–284 (1988).
3. R. A. Kaye, A Gray code for set partitions. *Information Processing Letters*, **5**, 171–173 (1976).
4. A. Nijenhuis and H. S. Wilf, *Combinatorial Algorithms*. Academic Press, New York (1978).
5. D. A. Patterson, C. H. Séquin, A VLSI RISC. *IEEE Computers*, **9**, 8–21 (1982).
6. I. Semba, An efficient algorithm for generating all partitions of the set $\{1,\ldots,n\}$. *Journal of Information Processing*, **7**, 41–42 (1984).
7. M. B. Wells, *Elements of Combinatorial Computing*. Pergamon Press, Oxford (1971).
8. D. Wilson, The Sun 4/260 RISC-based technical workstation. *Unix Review*, July 1988, 91–98; Japanese translation: *Unix Magazine*, **10**, 23–29 (1988).

Correspondence

A proposal for the renaming of the scientific branch of computer science

Dear Sir,

The creation, expansion and evolution of the scientific branch of Computer Science which is unique in world history is entirely due to the computer. The term *Computer Science* means, of course, the science of computers.

For this reason, perhaps, European scientists decided on the term *Informatique*, which was later accepted by nearly all the European countries. This term was derived from the combination of the French words *Information* and *Automatique*.

Personally I have been preoccupied with these terms, 'Computer science' and 'Informatique', because the first is based on the instrument (Computer), and the second on the kind of processing (Automatique).

In reality we have identified the machine or the kind of processing with the science itself. But our problem is, of course, the information itself and its change and not the computer or the kind of processing. It is as unsuitable to use the term 'Computer Science' as it would be to name, for example, Astronomy, 'Telescope Science', or for Microbiology to be called 'Microscope Science'.

For all these reasons we suggested in the late sixties the term 'pliroforiki' (πληροφορικη) to be adopted in Greece. This word is derived from the Greek work pliroforia (πληροφορια = information) with the suffix iki (-ικη). This suffix gives the word the precise meaning we need. That is, this term defines exactly the object of this modern science, which is the study of information and its change.

The richness and the exactness of the Greek language made it in the past a source of suitable terms for most of the scientific areas. Many of these terms are similar in form to the above term pliroforiki. Such examples are: 'logic' (λογικη), 'arithmetic' (αριθμητικη), 'mechanics' (μηχανικη), 'statistics' (στατιστικη), 'physics' (φυσικη), etc.

The term pliroforiki has been fully accepted in Greece in recent years. So I believe that an attempt should be made to establish this term internationally, because the term not only expresses completely the above mentioned branch of science, but is also consistent with corresponding terms of other scientific areas which have been established for many years. Hence the precise term and consequently its longevity is ensured.

I should be deeply obliged if you would publish this letter in your journal in order to invite wider discussions by specialists in this branch of science.

Yours faithfully

N. APOSTOLATOS
Od. Androutsou 7-Ag. Paraskevi,
Athens 15341, Greece

Editor's Comment

What's in a Name

In his letter, published above, Professor Apostolatos expresses a concern that neither of the phrases 'Computer Science' (English language) nor 'Informatique' (French language) reflects adequately the nature of the subject that the phrases are intended to represent. His contention is that the two phrases are too closely associated with computer hardware, whereas the central focus of computer-related studies is (or should be) information itself.

To my mind the principal weakness in the argument put forward by Professor Apostolatos is the contention that computer scientists should downgrade the importance of the tools with which they are working in relation to the medium. This seems akin to the suggestion that sculptors should rename their subject according to whether they are working with clay or bronze or wood or stone rather than in relation to the art of design. Information and information systems have been with us since the beginning of man's evolution; the tools for generating computer-based information systems have been with us only for some 45 years. Surely the thing that distinguishes the subject which we are considering is the emergence of the tools together with the techniques for

using them effectively and not the medium on which those tools are operating.

This is not to say that the time has not come for giving much closer attention to that medium than we have done in the past. In 1985 this *Journal* carried a set of special papers highlighting the need for such an approach, whilst the current issue is devoted largely to the study of information and its related disciplines. The focus of the special issue is emphasised by the final paper, written by Gordon Scarrott, in which he attempts to explore the nature of information and relate that nature to the wider aspects of computer science, but to say that we must place greater emphasis on the nature and role of information is not to say that we must discard full consideration of other things. As Gordon Scarrott states, the development of computerised judgement leads to the apparent need for neural nets; the two considerations (and others like them) go hand in hand.

Throughout its brief history our subject has been plagued with a plethora of names combined with little agreement on what those names should mean. In the early days the most common phrase in use was 'programming', although a programmer could be anybody from a numerical mathematician to a coder in a commercial organisation. Only when the development of programming languages and operating systems had reached the point where they could be studied in their own right did the phrase 'computer science' begin to make an appearance. Not everyone at that time, however, did use the phrase 'computer science'. Professor M. V. Wilkes, the first President of the British Computer Society, used for the title of his personal Chair at the University of Cambridge the term 'Professor of Computer Technology', whilst on the commercial front the 'data processing' industry was beginning to take off.

This raises the question of what exactly do we include within our subject area? Are computer applications not to be considered as a part of computer science? Were the early pioneers of EDSAC – who were carrying out text processing almost from the day that EDSAC began operations – not as involved in computer science as the later experts studying formal aspects of computing systems? There are many instances of important developments in computer science starting in the commercial data-processing world. The most far-reaching of these probably has been databases, which did not become accepted as a respectable academic subject until long after the work of the CODASYL Committee had made its presence felt in commercial data processing. The recent Bide Committee report (in the United Kingdom) has stressed the need for society to place as much emphasis on computer applications as on original research, and it appears inconceivable that we can consider as meaningful a subject area from which this particular aspect is excluded.

If our subject area is to consider the coming together of four entities – hardware, software, liveware and infoware (meaning information in all its guises, however it is coded symbolically) – and includes basic research, development and application in all these areas, then what generic term can we use to describe it? Some years ago the term information technology was introduced to emphasise the convergence of telecommunications with computing and to stress that online or distributed systems were now something to be considered seriously. Again, however, a title of this kind leads to an emphasis on components, tools and techniques rather than on the union of the whole subject area. Among other titles which are in common use are computer studies, computing studies, computing science, information science, information studies, information systems and information processing. The two phrases 'information science' and 'information studies' have always had different interpretations, the former being more generally associated with information retrieval rather than with computer science. But again, convergence is taking place, with multi-media integrated distributed office workstations making an appearance, generating the result that information science has become as much a part of 'computer science' as any of its other components. The other phrases are all, at some time, used with the same meaning.

Most recently has come the emergence of phrases such as software engineering, systems engineering, information engineering and information systems engineering. What do they mean? The phrases are intended to convey the idea that we should be building computer-based information systems using components, tools, standardisation, quality control, etc. Some of us may need to do this. But many of us are genuinely seeking fundamental knowledge about the nature of the environment in which we are working and, presumably, are scientists. Those of us who are anxious to explore the fundamental nature of information itself are, presumably, genuinely information scientists and not information engineers. Many others are concerned with the design of business systems or the application of computers in a commercial area such as art and design. None of these would consider themselves to be engineers. Other reasons have been quoted as to why the term 'engineer' is inappropriate (cf. the letter by C. R. Symons, *The Computer Bulletin*, Vol. 1, No. 1, February 1989). It is sufficient to say here that the term is not applicable in a large number of situations.

Whatever name is chosen there will be someone who will say that to them it means something else or that it does not cover adequately the whole subject area with which we are concerned. In the circumstances why do we not stop inventing new names and resurrect the one name which appears to cover all the different aspects of the subject area, that has been with us from the beginning, that has not been affected by the convergence of new academic areas or new technologies and that recognises the single factor which draws all the areas of study into one subject whole? This is the name 'computing'. For the foreseeable future, at least, this *Journal* will wish to be described as 'The International Journal of Computing' with all that that implies.

As Editor of The Computer Journal *I have been reluctant to be the one to comment on the letter by Professor Apostolatos in case comments which are entirely my personal view are taken to be, in some way, the official view of* The Computer Journal. *Would readers please note that, with one exception, the comment above is entirely personal and in no way reflects the views either of the Editorial Board of* The Computer Journal *or of the British Computer Society. The one exception is the final sentence.*

Peter Hammersley
Honorary Editor
The Computer Journal

Dear Sir,

In a recent paper published in your journal,[1] M. C. Er has shown that ordered trees with prescribed degree sequence can be generated by an elegant recursive algorithm that is simpler and more efficient than the algorithm of Zaks and Richards.[3]

However, we must point out that a superior algorithm with the same conceptual basis was published by Ruskey and Roelants van Baronaigien in 1984.[2] Our algorithm is reproduced below

```
procedure GenTree(lb, ub);
var i, j: integer;
begin
  if n_1 = n_2 = ··· = n_t = 0 then Print(a)
  else
    for i ∈ {lb, lb+1, ..., ub} do
      for j ∈ {p | p > 0 and n_p > 0} do begin
        n_j ← n_j − 1;
        a_i ← k_j;
        GenTree(i+1, ub+k_j);
        a_i ← 0;
        n_j ← n_j + 1;
      end;
end {of GenTree};
```

The sequences n_j and k_j have the same meaning as in Refs 1 and 3. The number of nodes with k_j children is n_j. Our sequence **a** corresponds exactly to Er's codewords *C*. To run our algorithm initialise **a** to be all zeroes and call *GenTree*(1, 1).

Let us call the algorithm of Ref. 1 *algorithm R*. The similarities of *GenTree* with algorithm *R* are self-evident. In fact, we had in the course of our research constructed exactly algorithm *R*, but later rejected it since it did not run in *constant average time* and it is slightly more complicated than *GenTree*. Constant average time means that the total amount of computation divided by the number of objects produced is bounded by a constant.

Algorithm *R* is not anlysed in Ref. 1. In Ref. 2 we show that the trees are generated in constant average time by *GenTree* if it is carefully implemented. The main idea is to maintain n_j and k_j as a linked list. In Ref. 2 we also show for the first time that permutations of a multiset can be generated in constant average time, and provide a general framework for showing that similar recursive algorithms operate in constant average time.

It is our opinion that every paper that is concerned with generating elementary combinatorial objects should at least consider the question of whether such generation is possible in constant average time.

yours faithfully,

FRANK RUSKEY
DOMINIQUE ROELANTS VAN BARONAIGIEN
Department of Computer Science,
University of Victoria,
P.O. Box 1700 Victoria, B.C.,
Canada, V8W 2Y2

References

1. M. C. Er, A simple algorithm for generating non-regular trees in lexicographic order. *The Computer Journal*, **31** (1), 61–64 (1988).
2. F. Ruskey and D. Roelants van Baronaigien, Fast recursive algorithms for generating combinatorial objects. *Congressus Numerantium*, 53–62 (1984).
3. S. Zaks and D. Richards, Generating trees and other combinatorial objects lexicographically. *SIAM J. Computing*, **8**, 73–81 (1979).