Correspondence

The Tower of Hanoi – Again

Sir,

M. C. Er's reply[1] to my letter[2] is a mixture of quibbles and personal abuse – a sure sign that someone has lost an argument.

The points he makes are easy to answer. Let us take them in order:

(1) The conclusion to be drawn from the work of Luger[3] is that the Tower of Hanoi is an easy problem when looked at the right way – how else can Er explain the 10.3% of subjects who solved the problem with a minimum number of steps on the **first** try? Like a good cryptic crossword clue it can seem extremely obscure until the right approach is found, and then seem extremely easy. It is this very property which makes the Tower of Hanoi a successful puzzle.

(2) Er claims that I constantly confused 'whether a problem is a trivial problem or whether a representation makes it so'. In the case of the Tower of Hanoi there is a representation – the tabulation of the states of the discs – which is trivially derivable and which makes the problem trivial. The two cases merge. Note that Er has nothing to say about the tabulation approach: it would destroy his case.

I agree with Er that representation approaches are valuable, but I say that not all representations are equally good. In choosing to represent moves rather than status, Er has ignored a simple representation, and one which has such close analogies with modular arithmetic and positional notation that deductions from it are easy and crisp. He has concentrated on moves and represented them as bit strings, for which mathematical apparatus is less well developed.

(3) Er's defence of the space he took to develop simple properties of the Tower of Hanoi is that other writers have taken even more space. This is a pretty poor defence, especially since most of the papers he cites were not attempting to solve the Tower of Hanoi problem, but were using it as an *example* in studying either human or machine problem solving.

(4) Since 0, 1 and 2 are merely tags given to the pegs in an arbitrary way, I lose no generality in always calling the source peg '0' and the peg to which the first move is made '1'. Assigning values to variables is so much the stuff of everyday Computer Science I thought no reader would have trouble with the concept. However, since Er has affected difficulty with renumbering, it is easy to transform the congruence to describe moves from any peg to any other.

If the source peg is s and the target peg is t, we can transform the original Pj to $P'j$ using the linear congruence:

$$P'_j \equiv s+(-1)^{t+1} P_j \pmod 3$$

I prefer the simplicity of renumbering.

(5) In criticising my proof, Er complains that I introduce peg Q and peg R 'without telling readers which one was peg 1 and which one was peg 2'. At that stage of the proof **it does not matter**. I need only to show how an optimal solution for discs 0, 1, ..., n can be derived from an optimal solution for discs 0, 1, ..., $n-1$.

Er's other point is answered by my remarks in section 4.

(6) Er attacks my derivation of his property 5, firstly by arbitrarily disallowing renumbering, and secondly by stating that the 'set of inequalities does not converge to 2^n-1; another valid solution is 2^n'. Since I said (and he quoted) the optimal solution occurs the *first* time the inequalities have a solution we can rule out 2^n as a solution.

(7) Er attacks my character and attitude because I state that his theorem connecting the Tower of Hanoi with the binary numerals follows immediately from the congruence. He states: 'Heard did not see it when he wrote his first correspondence or he would have mentioned it.' Er's theorem states that with 1-based disc numbering the number of 'the disc to be moved in step x is precisely the position index of the rightmost 1 in the binary numeral x'.

But the position index of the rightmost 1 in the binary numeral x is the position index of the digit which changes between the Gray code representation of $x-1$ and of x. Hence Er's theorem is exactly equivalent to the rule given by Martin Gardiner in 1972,[4] and quoted by Er in his original paper.

Why would I quote something already well known? And further, just what did Er think was new in his theorem?

I thank Er for his example of division using Roman numerals. It would be possible to write such an algorithm – it might be a good student exercise in string manipulation – but, because of the existence of arabic numerals, few would bother to write it up as a paper.

Given the existence of recursive and congruence solutions to the Tower of Hanoi problem, one cannot but reach a similar conclusion about Er's original paper.

Yours faithfully

R. J. HEARD

Queensland Institute of Technology,
G.P.O. Box 2434,
Brisbane,
Queensland,
Australia 4001

References

1. M. C. Er, The Tower of Hanoi problem – a further reply. *The Computer Journal* **28** (5), 543 (1985).
2. R. J. Heard, Further remarks on the trivial nature of the Tower of Hanoi problem. *The Computer Journal* **28** (5), 543 (1985).
3. G. F. Luger, The use of the state space to record the behavioral effects of subproblems and symmetries in the Tower of Hanoi problem. *International Journal of Man–Machine Studies* **8**, 411–412 (1976).
4. M. Gardiner, Mathematical games: the curious properties of the Gray code and how it can be used to solve puzzles. *Scientific American*, Vol. 227(2) 106–109 (1972).

A Biological Model Solution to the Towers of Hanoi Problem

Dear Sir,

Recent correspondence,[1,2] while mainly concerned with notions of complexity/simplicity specifically focused on the Towers of Hanoi Problem, at least emphasises the surprisingly many facets (solutions) to the problem. We say 'surprisingly many' because, since the solution is unique, only the mode of deriving it may vary. Further, both a recursive solution and a suggested iterative algorithm[3] were given at the very outset of the appearance of the problem (hereafter TH) in the literature (1892). Perhaps we shouldn't say 'surprisingly many' because in our mind we know very well that there is an infinite number of algorithms to solve any problem, yet in our heart we feel that all but a finite number are 'paddings' of the others. In any case, most of the solutions to TH have been variants on deriving iterative algorithms.[4,5,6,7] In this note we observe that the recursive solution is realised in a natural way by one of the simplest of the idealised biological growth models.

A DOL-system can be thought of as a finite sequence of cells each of which may 'divide', synchronously with the others, into a finite sequence of cells, according to certain rules. There is a collection S of states available to each cell: at each division, the daughter cells (and the states into which they are born) depend only upon the state of the mother cell at division. It is convenient to identify cells with their states, so that an array of cells is just a string over S, and the division rules are just productions $\{s \to W_s : s \in S\}$, where $W_s \in S^*$ (the set of finite strings over S). This is the simplest of the L-systems originated by A. Lindenmayer.[8] (A thoroughly mathematical treatment of L-systems is presented in Ref. 9.)

We choose some positive integer N and display a DOL-system which grows into the solution for TH for n rings ($n \leqslant N$). Let the pegs in TH be labelled A, B, C, and let the movement of a ring from X and Y be denoted by the doublet XY. Let $\mathscr{P}(A, B, C, n)$ be the procedure which transfers n rings from A to B, using C as an aid. The recursive solution for TH is just

$$\begin{aligned}\mathscr{P}(A,B,C,n) = {} & \mathscr{P}(A,C,B,n-1)\\ & \mathscr{P}(A,B,C,1)\\ & \mathscr{P}(C,B,A,n-1)\end{aligned}$$

Let $\mathscr{L}$ be a DOL-system in which the states are tuples $XYZk$ ($0<k<N$) and the production rules accord to the recursion above. That is, $XYZk \to XZY(k-1).XYZ1.ZYX(k-1)$ for $k>1$. (We use the period to indicate cell boundaries for the convenience of the reader.) For $k=1$, we write just XY in place of $\mathscr{P}(X, Y, Z, 1)$, since the procedure for a one-ring system is just the movement of the ring from A to B; the corresponding production is the identity $XY \to XY$ (at the next time-step the cell neither divides nor changes state). Let $\Rightarrow$ denote the transition of a cellular array from one time-step to the next. Note that each such transition represents the unfolding of a stage of the recursive solution to TH, and that, for an n-ring system, after n steps the cellular array is immobolised as the sequence of moves generated by $\mathscr{P}(A, B, C, n)$, where the cellular array begins as a single cell.

As an example, the growth of the system which represents the 3-ring TH is:

$$\begin{aligned}ABC3 &\Rightarrow ACB2.AB.CBA2\\ &\Rightarrow ABC1.AC.BCA1.AB.CAB1.CB.ABC1\\ &\Rightarrow AB.AC.BC.AB.CA.CB.AB\end{aligned}$$

where the last configuration is invariant.

Yours faithfully
R. M. BAER
Metadata Corporation
379 Countryview Drive
Mill Valley, CA 94941, USA

References

1. R. J. Heard, Further remarks on the trivial nature of the Tower of Hanoi problem. *The Computer Journal* **28** (5), 543 (1985).
2. M. C. Er, The Tower of Hanoi problem – a further reply. *The Computer Journal* **28** (5), 543–544 (1985).
3. W. W. Rouse Ball, *Mathematical Recreations and Essays*, Macmillan, London (1892).
4. P. Buneman and L. Levy, The Towers of Hanoi problem. *Information Processing Letters* **10**, 243–244 (1980).
5. P. J. Hayes, A note on the Towers of Hanoi problem. *The Computer Journal* **20** (3), 282–285 (1977).
6. T. R. Walsh, The Towers of Hanoi revisited: moving the rings by counting the moves. *Information Processing Letters* **15**, 64–67 (1982).
7. T. R. Walsh, Iteration strikes back – at the cyclic Towers of Hanoi. *Information Processing Letters* **16**, 91–93 (1983).
8. A. Lindenmayer, Mathematical models for cellular interactions. *Journal of Theoretical Biology* **18**, 280–315 (1968).
9. G. Rozenberg and A. Salomma, *The Mathematical Theory of L Systems*, Academic Press, New York (1980).

A Note on Parallel Topological Sorting

Dear Sir,

In his paper in *The Computer Journal*,[1] Er presents a topological sorting algorithm for parallel computers which he claims will take $O(D_{max})$ time, in the worst case, on SIMD computers with N processors (D_{max} is the longest distance from a source node to a sink node in the given directed graph). I intend to show that his result is weak and that some arguments he made in the paper are technically inaccurate.

(1) It is widely understood that in presenting an algorithm for a problem, one has to define precisely the computer model on which the algorithm is to be implemented. Er only briefly mentions that the computer model he used is an SIMD machine. Unfortunately, there are many types of SIMD machines, ranging from the unrealistic (theoretical) models, like WRAM and PRAM, to the realistic models, like the mesh-connected computers and the perfect-shuffle computers. The computation powers of these SIMD models vary significantly. For instance, for many graph-theoretic problems it is well known that there exist algorithms which run in $O(\log_2 N)$ time on the WRAM, whereas there is no algorithm which could run faster than $O(N)$ time on the mesh-connected computers. Therefore, simply stating that the computer model used is an SIMD is insufficient. Otherwise, it would be impossible to perform an accurate analysis on the time complexity.

(2) Data structures play an important role in algorithm design. Er does not specify what data structure he uses in storing the directed graph. He briefly mentions that each node of the digraph is stored in a separate memory module, but says nothing about how the edges are stored.

(3) Er observes that 'memory contention' (i.e. several processors attempt to write into the same memory location) could occur in his algorithm. He argues that as long as all those processors involved attempt to write the 'same' value into that memory location, then memory contention should not present any problem. This is because one of the processors can be chosen to perform the write operation while the rest are made inactive. This argument is certainly correct. However, he has forgotten to consider the time it takes to choose such a processor! (From the time complexity he claims in his paper, it is obvious that he has taken it for granted that this time is constant.) Since the set of processors involved in memory contention cannot be predetermined, all those processors involved have to communicate with each other in order to choose a representative. Undoubtedly, the time it takes to find a representative must be a function of p, where p is the number of processors involved. In the worst case, this time could be of order $\log_2 N$, where N is the number of processors in the SIMD machine. Therefore, Er has hidden at least a $\log_2 N$ factor under the constant factor of his time complexity.

(4) Er also neglects the processor-allocation problem. For instance, in Step 4 of his algorithm it is assumed that updating the values of all the N_s nodes takes $O(1)$ time. This is impossible, even in the case where the number of N_s nodes is $O(N)$ and there are N processors. The problem occurs because each N_p node has a different number of successor nodes. Although there are enough processors to handle the N_s nodes, it is not obvious to each processor which N_s nodes of which N_p node it is to handle. Resolving this processor-allocation problem could increase the time complexity by a factor of $\log_2 N$.

(5) The originality of Er's paper is also questionable. Er claims that his paper presents a 'new' topological sorting method based on a parallel computation approach. However, his algorithm turns out to be similar to the algorithm presented by Dekel, Nassimi and Shani.[2] Moreover, they implement their algorithm on two 'well-defined' SIMD models: the perfect-shuffle computers and the cube-connected computers. The time complexity they achieve is $O(\log_2 N^2)$ on both models, which is certainly better than Er's result.

Finally, it is worth noting that Dekel, Nassimi and Shani's algorithm can be implemented on a mesh-connected computer in $O(N)$ time with N^2 processors. This type of SIMD computers has certain attractive features: their structures are simple and the length of the wire connecting two processors is the same throughout the computer. As a result, they can be fabricated on a large scale using VLSI technology.

Yours faithfully,
YUNG H. TSIN
School of Computer Science
University of Windsor
Windsor, Ontario
Canada
N9B 3P4

References

1. M. C. Er, A parallel computation approach to topological sorting, *The Computer Journal* **26**, 293–295 (1983).
2. E. Dekel, D. Nassimi and S. Shani, Parallel matrix and graph algorithms, *SIAM Journal on Computing*, **10**, 657–674 (1981).