An Iterative Algorithm for the Reve's Puzzle

A divide-and-conquer approach for the presumed minimal solution to the Reve's puzzle, the Tower of Hanoi with four pegs, leads to a simple structure of the solution. In particular, it allows an easy, iterative algorithm for the three-peg problem—also extended to four pegs.

Received November 1989, revised November 1990

1. Introduction

The Towers of Hanoi, a familiar puzzle, is useful in discussing not only the principle of mathematical induction and the power of recursive algorithms, but also in discussing correctness proofs, the process of general problem solving, and answering the question: Which is better: Iteration or recursion? Herbert Simon studied the puzzle from a strategic viewpoint and showed that the difference between recursion and iteration can be brought back to the difference between demands on perceptual, short-term memory and long-term memory. Indeed, recursive solutions, elegant as they may be, do not readily show which step to take next. To alleviate this 'shortcoming', several iterative algorithms have been developed for the standard puzzle and for some of its variations. All of these iterative solutions have complicated correctness proofs, but all of these are based on the relation with the binary number system or with Gray codes. The following algorithm, already mentioned by Lucas, is the easiest to perform mentally (and manually), see also Hayes and Buneman and Levy. It is based on the fact that the smallest disc is moved at every odd move, and that only one legal move can be made at even moves.

A variation of the Towers of Hanoi, which allows for not three, but four pegs, was introduced by Dudeney in 1907 under the name of Reve's Puzzle. This problem—and the extension to any number of pegs—has been addressed by several authors. These papers, which address mainly the number of moves, imply a recursive solution where the tower is broken up in a number of subtowers. Llunnon pointed out that the offered solutions use an implicit assumption which is left unspecified. These solutions will subsequently be called presumed minimal solutions, pms.

Only recently, the first iterative algorithms for the Reve's Puzzle appear to have come from Lu and Hinz. The author of the first paper, Lu, noticed that it is possible to choose only three of the four pegs for completed subtowers. He then found a relation between the binary algorithm 3-peg Towers (n discs, from S, to D, using A) and

Algorithm 1: 3-peg Towers

Algorithm 2: 4-peg Towers

numbers and the controlling sequences which are in subsequently pre-computed. Hinz's algorithm first computes the number of subtowers and their bottom discs, after which it computes which subtower to move from which source peg to what target peg, and uses algorithm 1 for the actual moves.

In this paper, the solution is shown to have a very simple structure, allowing an easy computation for both the number of subtowers and the number of discs in a subtower. Furthermore, the solution can be cast in a form like algorithm 1, where the word 'disc' is replaced by 'slice', or 'subtower'.

2. An iterative solution

There are three phases in the standard solution to the 3-peg problem with n discs. In the first phase, all but the largest disc are moved to an auxiliary peg. This allows the remaining disc to be placed directly onto the target peg in the second phase. In the third phase, the discs on the auxiliary peg are moved to the target peg. These phases can be generalised immediately to the p-peg problem. In the first phase, a number of discs are moved to an auxiliary peg. This allows the remaining discs, assumed to be non-zero, to be placed onto the target peg in the second phase. Algorithm 1 can be made at even moves. The number of steps to be moved away during the first phase is thus not predetermined to be 'all but one'. In the third phase, the discs on the auxiliary peg are moved to the target peg.

The following algorithm 2 is given for the case that there are four pegs, the Reve's Puzzle. It differs from the one given by Reihl and Gedeon in that their function $s_1$, in their notation, is defined as $S(n) = \sqrt {2n + 1/4} - 1/2$ and has the property that $S(n)$ is strictly less than $n$ for all $n > 1$, which makes termination with $n = 1$ possible. The $S$-function in this paper, which is shown below to be the upper bound for the number of steps, is strictly less than $n$ only for all $n > 2$, so that algorithm 2 terminates with $n \geq 2$. Each level of the recursion causes the tower to be split up in two subtowers of sizes $(n-1)$ and $(n-2)$. The number of discs to be moved away during the first phase is therefore determined in advance to be 'all but one'. In the third phase, the discs on the auxiliary peg are moved to the target peg.

Algorithm 1: 3-peg Towers

Algorithm 2: 4-peg Towers

4. Concluding remarks

There are three phases in the standard solution to the 3-peg problem with $n$ discs. In the first phase, all but the largest disc are moved to an auxiliary peg. This allows the remaining disc to be placed directly onto the target peg in the second phase. In the third phase, the discs on the auxiliary peg are moved to the target peg.

The number of moves in a pms for the Reve's puzzle is given, by Hinz, as $T(n)$

$$T(n) = \min \{2T(n-1) + 2 + T(\lfloor \frac{n}{2} \rfloor) \}$$

where the min is taken such that $0 < s_1 < s_2 < \ldots < s_n = n$. Equation (1) has two counteracting forces at work: if the number of slices gets large, then the term with two $s$s dominates, whereas if $n$ is small, then at least one of the slices is large (i.e. $s_1$ is large). In a presumed minimal solution, all the powers of two are equal or almost equal, so that $s_i = c + i$ for all $i$ and constant $c$: the $s_i$'s form a strictly decreasing sequence with two consecutive slices differing by at least one and at most two discs. Furthermore, at most one pair of slices that differ by two. These restrictions determine the size of the bottom slice $s_1$; if all $s_j - s_{j+1} = 1$ for all $i$, then the stack is 'saturated'; any increase in stacksize would necessitate an increase in the number of slices. But this means that $n = m + 2 + 1$, i.e. $n$ is a binomial number, or $n = \sqrt{2n + 1/4} - 1/2$. In this case $s_j$ is the only slice which is not equal to $s_i$; so that $s_j = \sqrt{2n + 1/4} - 1/2$. The sizes of subsequent slices can be found recursively, but, by the above, $s_j$ differs from $s_i$ by either one or two: $s_j$ equals either $s_i - 1$ or $s_i - 2$. The process can be continued, and, this way, every $n$ can be written uniquely as such a sequence of slices $s_1, s_2, \ldots, s_m$, although these computations are rather cumbersome to perform mentally. Table I shows both the slice-sequence and the pms for the first few numbers. An alternative version of the recursive 4-peg algorithm can be obtained by using automatic recursion removal. Alternatively, one could try to extend any interactive scheme for the 3-peg problem. Algorithm 1, where moves alternate between moving the smallest disc and the other possible move, is particularly well suited for this task: Alternate between the smallest slice and the other possible slice movement. In the 3-peg problem, all the moves are well-defined. The smallest disc moves in a cyclic fashion to the other numbered moves, and there is only one other move possible at even-numbered moves. In the 4-peg problem, all discs get to move in a cyclic fashion to the other numbered moves, and there is only one other move possible at even-numbered moves. In the 4-
The algorithm, given below, shows the move instructions for the slices, but not for the individual discs. Each slice movement is however the traditional 3-peg Tower of Hanoi and has been omitted here. Its correctness follows from the fact that it starts with all slices stacked on top of each other on one peg S, makes legal moves with all slices stacked on top of each other on one peg D. Furthermore, within a slice, the placement rule is never violated. Finally, the number of moves generated by the algorithm are those generated by Hinz’s algorithm, as $s_j = n(n)$ is a minimum partition number for $n$. $T_3(n) = \min (2^{n(n-1)}+2^{n-1})$.

3. Conclusion

The above sketched method suggests an iterative algorithm with a distinctive recursive flavor for the multi-peg problem with more than four pegs. Indeed, the solutions as given by Frame, Stewart and Brousseau follow this pattern. However, for the 3-peg problem, I have failed to obtain an optimal sequence of slices $u_j$ so that each slice can be solved with the 4-peg solution and such that all slices add up to $n$. Notice that the iterative algorithms of Hinz’s algorithm—width a distinctive recursive flavor—for the multi-peg problem with more pegs.

Algorithm Iterative Reve’s ($n$ dissc, from $S$, to $D$, using $A_1$ and $A_2$)

(*Determine the sizes of the slices*)

$m = 0$

for $j$ from $1$ to $m$ do $s_j = i + 1$ endfor

for $j$ from $1$ to $m$ do label disc $s_j$ with label $i$ endfor

(*Determine the cyclic order for the three main pegs*)

if ($m$ is even) then cyclic order is $S$ to $A_2$ to $D$ to $S$ to...

else cyclic order is $S$ to $D$ to $A_1$ to $S$ to...

(*Make the actual moves*)

move smallest slice (highest label) in cyclic order, using all 4 pegs.

until done do

move the slice with next highest labels, bypassing the peg that contains the smallest slice.

move smallest slice in cyclic order, using all 4 pegs.

end until

Algorithm 3: Iterative Reve’s

A. Van de Liefvoort

References