The Generalized Colour Towers of Hanoi: An Iterative Algorithm

M. C. Er
Department of Computing Science, The University of Wollongong, P.O. Box 1144, Wollongong 2500, N.S.W., Australia

An iterative algorithm for solving the generalized colour Towers of Hanoi problem is presented; and its underlying principles are discussed. The problem is a variant of the Towers of Hanoi problem; it has $n$ black and white discs randomly stacked on three pegs as an initial configuration. The objective is to move all coloured discs to a specified peg subject to the usual constraints of the standard problem; in addition, white and black discs may only move clockwise and counterclockwise, respectively. A comparison with a recursive algorithm for solving the same problem is also made.

INTRODUCTION

The Towers of Hanoi is well known as a complex problem which has remarkably simple solutions. Many interesting but different solutions, both recursive and iterative, have been discovered.\(^1\)\(^-\)\(^5\) The interest in the Towers of Hanoi seems to have shifted from the standard problem to its variants. Atkinson\(^6\) suggests the cyclic Towers of Hanoi problem and a recursive solution to it; Er\(^7\) describes an iterative solution which is remarked by Atkinson as a non-trivial exercise. Er\(^8\) also proposes the generalized Towers of Hanoi problem, and presents a solution to it. Among many other interesting variants, Er\(^9\) describes one of the most intricate yet challenging problems—the generalized colour Towers of Hanoi problem—and presents a simple recursive solution to it.

The generalized colour Towers of Hanoi problem may be summarized briefly as follows. There are three pegs and $n$ discs of different sizes, each of which is coloured either black or white. The pegs are arranged as a circle so that the clockwise and the counterclockwise directions may be defined in the usual sense when viewing from the top. The discs are stacked on one or more pegs initially in increasing sizes with the largest discs at the bottom. The task is to move all discs to a specified peg subject to the following constraints:

1. Only one of the topmost discs may be moved at a time.
2. No disc may ever be stacked on a disc smaller than itself.
3. White and black discs may be moved to their neighbouring pegs only in the clockwise and the counterclockwise directions respectively.

Solutions to this generalized colour problem are complicated by the two added factors which are not present in the standard problem:

(i) The initial configurations of discs on the three pegs are purely random.
(ii) The combinations of colours, which are again random, may hinder the moves of some discs.

If we now add a requirement of ensuring the minimum number of disc moves is taken to attain the goal for any given configuration of coloured discs, then solutions to this generalized colour Towers of Hanoi problem are truly non-trivial.

In this paper, we develop an iterative solution to this generalized colour problem and discuss the underlying principles. A recursive solution to this complex problem employing different strategies can be found in Ref. 9.

UNDERLYING PRINCIPLES

In this section, we establish a few principles which are necessary and crucial to the derivation of the iterative solution.

As noted in the previous section, the initial configurations of coloured discs are purely random; and hence the first move of each disc is very much dependent on the given configuration and is not an \textit{a priori} of the solution. However, when more and more discs are moved in the course of attaining the goal, a regular pattern of disc moves gradually emerges; and this is no doubt due to the following trivial observation.

Theorem 1

When a disc is moving from a source peg to a target peg, all discs smaller than this disc must be stacked on a spare peg.

\textbf{Proof.} The proof follows readily from the constraints of the game. Owing to Constraint C1, the disc about to move must be exposed as a top disc; and hence none of the smaller discs could stack on the source peg. Further, none of them could stack on the target peg either; otherwise Constraint C2 will be violated when the disc in question is moved. Summing up, the smaller discs must stack on a remaining peg, a spare peg, when a disc is moving from a source peg to a target peg.

When all discs smaller than a given disc, forming a subtower on a peg, are to move to another peg, a sequence of disc moves for attaining this subgoal is a regular pattern and is determined by the state of the disc they are smaller than. The regularity is also influenced by the combination of colours of the discs forming the subtower. It is important to study this regularity in order to avoid treating each disc move as a special case.
When a disc is to move to the next destination, one of two possibilities must exist—the destination can be reached in either one step or two steps depending on the combination of disc colour and the destination to move to; we call these two cases the one-step and the two-step contexts, respectively. There are no other contexts besides these two; either they can be reduced to one of these two simple forms or they represent no progress made which is impossible. Let \( C_m(1) \) and \( C_m(2) \) denote the one-step and the two-step contexts, respectively, of moving disc \( m \). The disc moves and their contexts may be formalized as follows.

**Theorem 2**

When a subtower of \( m \) coloured discs is transferred to another peg, the sequence of disc moves is given by one of the following context equations:

\[
\begin{align*}
C_m(1) &= C_{m-1}(1)mC_{m-1}(1) \\
C_m(2) &= C_{m-1}(2)mC_{m-1}(1) \\
C_m(1) &= C_{m-1}(1)mC_{m-1}(2) \\
C_m(2) &= C_{m-1}(1)mC_{m-1}(2)
\end{align*}
\]

Here, the \( m \) written independently represents moving disc \( m \) one step in the direction appropriate to its colour; and

\[
\begin{align*}
C_1(1) &= 1 \\
C_1(2) &= 1 \ 1
\end{align*}
\]

**Proof.** Note that disc \( m \) may move in the one-step or the two-step contexts and that discs \( m \) and \( m - 1 \) may have the same or different colours; so the combination gives rise to four distinct cases.

Case (a):
Disc \( m \) moves in the one-step context and its colour is different from that of disc \( m - 1 \). Before disc \( m \) could be moved, disc \( m - 1 \) and all smaller discs, if any, should be cleared to a spare peg by Theorem 1; then disc \( m \) is moved one step to a target peg; and finally disc \( m - 1 \) and all smaller discs are moved again to the target peg. In both instances, disc \( m - 1 \) is moved in the one-step context because it has a different colour from disc \( m \) which is also moved in the one-step context. Thus, Eqn (1) is correct.

Case (b):
Disc \( m \) moves in the one-step context and its colour is identical with that of disc \( m - 1 \). The sequence of disc moves is similar to Case (a) but the two instances of the moves of disc \( m - 1 \) should be in the two-step context instead. This is because both discs \( m \) and \( m - 1 \) have same colour; and thus when disc \( m \) is moved in the one-step context, disc \( m - 1 \) cannot be moved in the same context. Hence Eqn (2) holds.

Case (c):
Disc \( m \) moves in the two-step context and its colour is different from that of disc \( m - 1 \). To transfer the subtower from a source peg to a target peg with the minimum number of disc moves, again by Theorem 1, disc \( m - 1 \) and all smaller discs should be moved first to the target peg before disc \( m \) is moved one step to a spare peg appropriate to its colour, then back to the source peg before disc \( m \) is moved again one step to the target peg, and finally to the target peg again to complete the task. Since discs \( m \) and \( m - 1 \) have different colours, the moves of disc \( m - 1 \) from the source peg to the target peg should be in the one-step context; and conversely, the moves of disc \( m - 1 \) from the target peg to the source peg must be in the two-step context. Thus Eqn (3) is also true.

Case (d):
Disc \( m \) moves in the two-step context and its colour is identical with that of disc \( m - 1 \). The argument is similar to Case (c), but the moves of disc \( m - 1 \) are in the opposite contexts instead. The same colour of these two discs compels disc \( m - 1 \) to move from the source peg to the target peg in the two-step context and the reverse process in the one-step context as disc \( m \) can only move one step at a time to the next appropriate pegs. Equation (4) is thus proved correct.

Since there are no other combinations, the sequence of disc moves for transferring a subtower to another peg must be given by one of the above cases.

Theorem 2 could be applied recursively, and thus be used in deriving the following important result.

**Theorem 3**

The context-change may be stated in two parts:

(i) When a disc makes a move in the one-step context, the next smaller disc need not change its context.

(ii) When a disc makes a move in the two-step context, the next smaller disc changes its context, either from one-step to two-step or from two-step to one-step.

**Proof.** We prove the theorem part by part.

(i) From Theorem 2 (1) and (2), we see that when disc \( m \) moves in the one-step context, the contexts for moving disc \( m - 1 \) before and after the move of disc \( m \) are identical. By induction, the first part is correct.

(ii) As the name implies, a disc moving in the two-step context ought to move two single steps according to its colour. From Theorem 2 (3) and (4), we may see that every time disc \( m \) makes a move, the context for moving disc \( m - 1 \) changes its parity. Again by induction, the second part is also true.

From Theorem 2, we see that \( C_1(1) = 1 \) and \( C_1(2) = 1 \ 1 \). By treating independent moves and consecutive moves of the smallest disc as packets, we may state a regularity of disc moves as follows.

**Theorem 4**

Packets of moves of the smallest disc alternate with moves of other discs.

**Proof.** In the generalized colour problem, the smallest disc may not be the first disc to move; but this does not affect our argument. Consider a general case. Suppose all three pegs are occupied by discs; then one of them must have the smallest disc as the topmost disc by constraint C2. We further assume that the smallest disc is the next disc to move. It may be in the one-step or the two-step context, and will be moved accordingly. After the smallest
disc is moved, the smaller topmost disc of the two remaining pegs not occupied by the smallest disc must be the next disc to move. Under this situation, one of two things must happen: either a move of the smaller disc will stack it on top of the smallest disc thus violating constraint C2 or the smaller disc could be validly moved to another peg not occupied by the smallest disc. In the former case, the illegal move cannot be carried out; and the next disc to move is the smallest disc—such successive moves obviously could be replaced by a packet of one or two moves. Thus we have only the latter case to be considered. After the smaller disc moves one step, it cannot move any further without stacking on the smallest disc due to the colour constraint, nor may other discs besides the smallest disc be moved. In consequence, we are cyclically back to the original state.

The other special cases such as one of the two pegs not occupied by the smallest disc is empty or both are empty may also be easily proved. In the former case, the smaller disc is also unique, and the situation is similar to the general case. In the latter case, we have trivially moved all discs to a peg.

If the consideration starts from the move of other discs but not the smallest disc, the argument is cyclically similar.

By induction, we thus complete the proof.

Now, we make explicit a principle which is implicitly stated in Theorem 2.

**Theorem 5**

In transferring a subtower of \( m \) coloured discs from one peg to another peg, the *first* move of disc \( i \), \( i < m \), is in the one-step context, before disc \( m \) makes its first move, if discs \( i \) and \( i+1 \) are of different colours, and is in the two-step context otherwise.

**Proof.** From Theorem 2, the context equations governing the first move of disc \( m-1 \) are Eqs (1) and (3) if the colours of discs \( m \) and \( m-1 \) are different. It is obvious that, from these two equations, the first move of disc \( m-1 \) is in the one-step context before the first move of disc \( m \) irrespective of which context disc \( m \) is in. A similar argument could also be established when the colours of both discs are identical by applying Eqs (2) and (4); in this case, disc \( m-1 \) moves in the two-step context instead. By induction on the smaller discs of the subtower, the theorem is correct.

The above theorem is useful when the colour Towers of Hanoi problem starts with only one tower of discs. To tackle the generalized colour problem with more than one tower, we need to make use of the following result.

**Theorem 6**

In transferring a subtower of \( m \) coloured discs from one peg to another peg, the *last* move of disc \( i \), \( i < m \), just before disc \( m \) makes its first move is in the one-step context if discs \( i \) and \( i+1 \) are of different colours, and is in the two-step context otherwise.

**Proof.** The proof relies on the symmetry argument. From Theorem 2, one may see that the four context equations are symmetric; and thus the first move and the last move of disc \( m-1 \) are in the same context given by a context equation of disc \( m \). If the argument is applied recursively, this theorem can be derived from Theorem 5 by replacing ‘the first move of disc \( i' \) with ‘the last move of disc \( i' \).

**INITIALIZATION AND TRANSITION**

In the previous section, we have established a few broad principles which govern the regularities of disc moves in transferring a subtower of coloured discs to a target peg. However, as pointed out before, the initial state of a given configuration of coloured discs is purely random; and therefore the transition from a random state to a more regular state is of critical importance and should be well planned. Further, the initial context for moving each disc needs to be properly initialized so that a smooth transition can take place. We address these two important issues in this section, first the initialization phase.

The purpose of the initialization phase is to determine the initial context for making the first move of each disc. The first move of a disc is influenced by the first move of the next disc larger than itself; in turn, the first move of the largest disc is determined by the goal to be attained. So, the initial context of a disc is completely determined by its colour, its home peg and the next destination to move to, and may take one of the three forms—zero-step, one-step and two-step contexts. If the next destination could be reached in one step according to the disc colour, the one-step context is assigned. If, however, the next destination could only be reached in two steps, owing to the colour constraint, the two-step context is given. Also, it may well be that a disc already resides on the destination dictated by the next larger disc or the goal, as appropriate; in this case, the zero-step context is assigned to this disc so that the implicit ‘first move’ need not be carried out. Clearly a disc being assigned the zero-step context exerts no influence on the initial context of the next smaller disc. In determining the next destination a disc should move to, we note that the dominating disc (the next larger disc which is not assigned the zero-step context) can only move to the next appropriate peg according to its colour; and therefore the next destination must be the peg untouched by the dominating disc during its first move.

During the transition from a random state to a more regular state, when a disc is about to make its first move, all smaller discs must stack on a spare peg forming a subtower by Theorem 1. At this moment, the contexts of all smaller discs can be enforced as per Theorem 6 as if they have just been moved away from the disc about to make its first move. Thus if the first moves of all discs, except those assigned the zero-step context, are accounted for in this way, a smooth transition can be achieved. More importantly, such a smooth transition guarantees that the minimum number of disc moves is taken to attain the goal.

**ITERATIVE ALGORITHM**

Assuming the following constant specifications:

- \( \text{nulldisc} = \maxint \);
- \( \text{zerostep} = 0 \);
- \( \text{onestep} = 1 \);
- \( \text{twosteps} = 2 \);
and the following type definitions:

\[
\begin{align*}
\text{peg} & = 1 . . 3; \\
\text{discs} & = 1 . . \text{maxdisc}; \\
\text{nulldisc} & = 0 . . \text{maxdisc}; \\
\text{colourdisc} & = -\text{maxdisc} . . \text{maxdisc}; \\
\text{step} & = 0 . . 2;
\end{align*}
\]

we may declare the following global arrays:

\[
\begin{align*}
\text{sourcepeg} & : \text{array} [\text{discs}] \text{of peg}; \\
\text{nextdisc} & : \text{array} [\text{discs}] \text{of integer}; \\
\text{topdisc} & : \text{array} [\text{Peg}] \text{of integer}; \\
\text{disc} & : \text{array} [\text{discs}] \text{of colourdisc};
\end{align*}
\]

such that \text{sourcepeg} \{d\} contains the source peg of disc \text{d},
\text{nextdisc} \{d\} contains the disc beneath disc \text{d} or the value \text{nulldisc} if disc \text{d} is a bottom disc;
\text{topdisc}[\text{p}] contains the topmost disc on peg \text{p} or the value \text{nulldisc} if no disc is on the peg;
and \text{disc}[d] contains the integer representation of disc \text{d}—positive and negative integers for white and black discs respectively, with larger discs being assigned large absolute numbers. These global arrays are initialized according to a given configuration of coloured discs.

An iterative solution to the generalized colour Towers of Hanoi problem may be described by the following Pascal-like program.

```
procedure IterGenColourTowers (n: \text{movefolders}; target: \text{peg});
{ When called with the total number of coloured discs stacked on the pegs, this procedure moves all of them to the target peg subject to the stated constraints. It assumes that the global arrays \text{sourcepeg}, \text{nextdisc}, \text{topdisc} and \text{disc} have been initialized according to a given configuration of coloured discs. }

\text{var} \quad d, i : \text{discs};
\text{pa, pb : peg};
\text{movestep : array} [\text{discs}] \text{of step};
\text{firstmove : array} [\text{discs}] \text{of boolean};
\text{begin}
\text{if} \ n > 0 \ \text{then begin}
\text{for} \ d := n \ \text{downto} \ 1 \ \text{do begin}
\text{movestep}[d] := \text{WhatStep}(\text{disc}(d), \text{sourcepeg}(d), \text{target});
\text{if} \ \text{movestep}[d] < \text{zerostep} \ \text{then begin}
\text{firstmove}(d) := \text{true};
\text{target} := \text{NextTarget}(\text{disc}(d), \text{sourcepeg}(d))
\end{begin}
\text{end};
\text{end};
\text{if} \ \text{firstmove}(1) \ \text{then MoveSmallestDisc (movestep(1))};
\text{loop}
\text{pa} := \text{NextPeg} (\text{sourcepeg}(1));
\text{pb} := \text{NextPeg} (\text{pa});
\text{if} \ \text{topdisc}[\text{pa}] = \text{topdisc}[\text{pb}] \ \text{then return};
\text{if} \ \text{topdisc}[\text{pa}] < \text{topdisc}[\text{pb}]
\text{then} \ d := \text{topdisc}[\text{pa}]
\text{else} \ d := \text{topdisc}[\text{pb}];
\text{if} \ \text{firstmove}(d) \ \text{then begin}
\text{firstmove}(d) := \text{false};
\text{for} \ i := d - 1 \ \text{downto} \ 1 \ \text{do}
\text{if} \ \text{IsWhite}(\text{disc}(i + 1)) = \text{IsWhite}(\text{disc}(i))
\text{then movestep}(i) := \text{twosteps}
\text{else movestep}(i) := \text{onestep};
\text{end};
\text{MoveDisc(sourcepeg(d));}
\text{if} \ \text{movestep}(d) = \text{twosteps} \ \text{then movestep}(d - 1) := 3 - \text{movestep}(d - 1);
\text{MoveSmallestDisc(movestep(1))}
\text{repeat}
\text{(IterGenColourTowers)};
\text{end;}
```

The function \text{WhatStep} computes the initial context of a disc for making its first move, and may be described as follows:

```
function \text{WhatStep}(d: \text{colourdisc}; source, target: \text{peg}): \text{step};
{ This function determines the initial context of disc \text{d} from its colour, source peg and the target peg it should move to. }
\text{begin}
\text{if} \ \text{source} = \text{target} \ \text{then WhatStep} := \text{zerostep}
\text{else if} \ \text{IsWhite}(d) = \text{IsClockwise}(source, target)
\text{then WhatStep} := \text{onestep}
\text{else WhatStep} := \text{twosteps}
\text{end} \ (\text{WhatStep});
```

Here, the predicate \text{IsWhite} \((d) = \text{IsClockwise}(source, target)\) is true only when a white disc is to move clockwise or when a black disc is to move counterclockwise. These two functions may be described as follows.

```
function \text{IsWhite} (d: \text{colourdisc}; source: \text{peg}): \text{boolean};
\text{begin}
\text{IsWhite} := d > 0
\text{end};
```

```
function \text{IsClockwise} (source, target: \text{peg}); \text{boolean};
\text{begin}
\text{IsClockwise} := (\text{target} - \text{source} = 1) \text{or} (\text{target} - \text{source} = -2)
\text{end};
```

The function \text{NextTarget} determines the destination the next smaller disc should move to from a given disc and its home peg. The destination is precisely the next peg the given disc would move to had it changed to the opposite colour.

```
function \text{NextTarget} (d: \text{disc}; source: \text{peg}); \text{peg};
{ Compute the target peg the next disc smaller than disc \text{d} resting on a source peg should move to. }
\text{begin}
\text{if} \ \text{IsWhite}(d)
\text{then NextTarget} := \text{PrevPeg(source)}
\text{else NextTarget} := \text{NextPeg(source)}
\text{end} \ (\text{NextTarget});
```

Here the functions \text{PrevPeg} and \text{NextPeg} compute the previous peg and the next peg respectively with respect to the clockwise direction as follows:

```
function \text{PrevPeg} (p: \text{peg}); \text{peg};
\text{begin}
\text{PrevPeg} := (p + 1) \mod 3 + 1
\text{end};
```

```
function \text{NextPeg} (p: \text{peg}); \text{peg};
\text{begin}
\text{NextPeg} := p \mod 3 + 1
\text{end};
```

The procedure \text{MoveSmallestDisc} moves the smallest disc one or two steps in the direction consistent with its colour depending on the context it is in as follows:

```
procedure \text{MoveSmallestDisc} (s: \text{step});
\text{begin}
\text{MoveDisc (sourcepeg(1));}
\text{if} \ s = \text{twosteps} \ \text{then MoveDisc (sourcepeg(1))}
\text{end} \ (\text{MoveSmallestDisc});
```

The procedure \text{MoveDisc} simply moves the topmost disc on a given peg by updating the global arrays as follows:

```
procedure \text{MoveDisc} (p: \text{peg});
\text{var target : peg;}
\text{discomoved : discs;}
\text{begin}
discomoved := \text{topdisc}(p);
\text{if} \ \text{IsWhite}(\text{discomoved})
\text{then target} := \text{NextPeg}(p)
\text{else target} := \text{PrevPeg}(p);
\text{topdisc}[p] := \text{nextdisc}[\text{discomoved}];
\text{nextdisc}[\text{discomoved}] := \text{topdisc}(\text{target});
\text{topdisc}(\text{target}) := \text{discomoved};
\text{sourcepeg} [\text{discomoved}] := \text{target}
\text{end} \ (\text{MoveDisc});
```

Note that the predicate \text{IsWhite}(\text{disc}(i + 1)) = \text{IsWhite}(\text{disc}(i)) is true only if discs \(i\) and \(i + 1\) are of similar colours, capturing the essence of Theorem 6.
CORRECTNESS PROOF

To prove informally that the iterative algorithm discussed in the previous section is correct, we need to show that the following four conditions are satisfied:

(a) Packets of moves of the smallest disc alternate with moves of other discs.
(b) The context array moveStep is updated correctly.
(c) The context array moveStep is used correctly.
(d) The terminating condition is both necessary and sufficient.

The correctness of condition (a) may be easily seen from the procedure IterGenColourTowers. Within the last control loop, the calls to procedure MoveSmallestDisc to move the smallest disc and to procedure MoveDisc to move other discs alternate in sequence. Thus Theorem 4 is satisfied. Further, the procedure always chooses the smaller disc from the two pegs not occupied by the smallest disc during the alternate move; so a disc would not stack on a disc smaller than itself.

To show that the context array moveStep is updated correctly, we need to verify its initialization, transition from a random state to a more regular state, and context change during each disc move. As may be seen from the procedure IterGenColourTowers, the first for-loop implements the ideas of initialization correctly as discussed in Section 3. Further, a Boolean array firstMove is also initialized to keep track of the first move of each disc, which ought to be actually carried out during a transition. And as seen in the same procedure, the essence of Theorem 6 is enforced upon the smaller discs by adjusting their contexts when such a transition is about to take place. Finally, we see that when a disc other than the smallest disc is moved, the context of the next smaller disc is changed in accordance with Theorem 3. Thus the context array moveStep is maintained correctly.

The only place using the array moveStep to move a disc is the procedure MoveSmallestDisc. It may be easily seen that the smallest disc is moved one or two steps according to its context. The smallest disc is a dominating factor in the iterative solution; it influences and uniquely determines the smaller discs in the alternate moves. Paradoxically, the context is propagated from larger discs to smaller discs and eventually upon the smallest disc. So if the propagation of context is carried out correctly as verified above, the smallest disc will move in the right manner.

Lastly the terminating condition is also easy to verify. When two pegs are empty, all discs must have stacked on a third peg already. Further, a sequence of disc moves for transferring a tower of coloured discs to a specified peg is finite; if all discs are moved correctly as discussed above, the terminating condition will be met in a finite time. Hence the terminating condition is both necessary and sufficient.

CONCLUDING REMARKS

Since an iterative algorithm and a recursive algorithm for solving the generalized colour Towers of Hanoi problem are available, it is interesting to compare their performance. An empirical testing reveals that both algorithms take the same sequences of disc moves in transferring all coloured discs given in any arbitrary configurations to specified pegs; but the iterative algorithm takes less time than the recursive algorithm to complete the task. The difference in time-complexity is due to the use of different underlying principles and strategies in deriving the two algorithms, and this is reflected in the computations. Regarding the space-complexity, the iterative algorithm uses $O(6n)$ storage; in contrast, the recursive algorithm uses $O(3n)$ storage excluding the recursion stack.

In conclusion, we have derived an iterative algorithm for solving the seemingly complex problem—the generalized colour Towers of Hanoi problem, and have discussed at length its underlying principles. If the colours of the discs are restricted to white, then we have the generalized cyclic Towers of Hanoi problem. If the number of initial towers is further restricted to one, we have the cyclic Towers of Hanoi problem. Clearly these two variants are special cases of the general problem discussed in this paper; and thus the iterative algorithm could be used to solve with ease.

Acknowledgement

This research was supported by RGC under grant 05-143-105.

REFERENCES


Received March 1983