Design and Implementation of an Efficient Natural Algorithm to Solve Towers of Hanoi Puzzle having Multiple Towers & Disks

A. S. Zaforullah Momtaz, Md. Fayyaz Khan, Muhammad Sajjad Hossain, Aloke Kumar Saha and Kazi Shamsul Arefin

***Abstract*—Many algorithms are available to solve the puzzle. Natural Algorithm (NA) developed and presented in this paper not only provides solution for the three tower and eight disks problem, but provides solution for multiple towers and multiple disks problem through minimum number of operations. Solution for sixteen towers and fifteen disks has been shown in the paper requiring minimum number of operations.**

***Index terms*—Natural Algorithm (NA), Towers of Hanoi (TOH), N-Tower Solution.**

I. Introduction

Towers of Hanoi (TOH) are a familiar mathematical puzzle introduced by a French mathematician Edouard Lucas in 1873 [1][3]. The puzzle deals with three towers and eight disks of different sizes. Initially all the disks are placed randomly in a tower called the source tower. All these disks from the source tower will have to be moved to another tower known as target tower with the help of an intermediate tower called the "rest tower". The disks are to be placed in the target tower according to the size of the disk i,e the largest disk will occupy the bottom position in the stack of the target tower and the rest of the disks are to be placed uniformly in the descending size of the disk. Hence the smallest disk will occupy the top most position in the stack of the target tower. Minimum number of operations will be required to transport disks from the source tower to the target tower [7-11].

The TOH puzzle has two major parts. One of them is the number of movements of the disks for a particular combination of tower & disk, and the other one is the movements of the disks between the source towers to the target tower [4].

When there are 3 towers in the puzzle, then the solution is quite straight forward [5]. But with the increase of towers, complexity of the problem increases manifold that involves significant number of equations and the algorithm size increases appreciably. In this paper, natural algorithm has been developed that provides solution to the tower of Hanoi (TOH) problem with finite number of towers and disks.

Manuscript received February 2, 2011

A. S. Zaforullah Momtaz is with the Department of Computer Science and Engineering, University of Asia Pacific (www.uap-bd.edu), Dhanmondi, Dhaka-1209, Bangladesh.

Md. Fayyaz Khan is with the Department of EEE, United International University, Dhanmondi, Dhaka-1209, Bangladesh.

Muhammad Sajjad Hossain is serving in the Department of Arts and Sciences, Ahsanullah University of Science and Technology, Dhaka-1208, Bangladesh. E-mail: www.aust.edu

Aloke Kumar Saha is with the Department of Computer Science and Engineering, University of Asia Pacific (www.uap-bd.edu), Dhanmondi, Dhaka-1209, Bangladesh. E-mail: aloke@uap-bd.edu.

Kazi Shamsul Arefin is with the Department of Computer Science and Engineering, University of Asia Pacific (www.uap-bd.edu), Dhanmondi, Dhaka-1209, Bangladesh. E-mail: arefin@uap-bd.edu.

II. Natural Function

In the puzzle, there are two terms- tower and disk. For a specific number of towers the number of disks may vary. It may be above, or below, or equal to the number of towers. Depending on these two variables, different equations are obtained for the natural function. Let, t and d are two variables namely towers and disks respectively. These two are defined as $(0 < t \leq n)$ and $(0 < d \leq m)$ where n and m represents the highest number of the towers and disks respectively. Thus we have the terms t, d & N and t, d > 0. Using these two variables, the 'Natural' function is implemented to find out the total no. of movements. Here 'm' is considered as 15 (m=15). For this limitation the function is defined as,

$$f(t, d) = \begin{cases} 2^d - 1 & (\text{for, } t = 3) \\ 2d - 1 & (\text{for, } t > d) \\ 4d - 2t + 1 & (\text{for, } t \leq d \leq (t\,(t-1)) / 2) \\ 2t^2 - 4t + 9 & (\text{for, } d = (t\,(t-1)) / 2 + 1) \\ 20 + 2f(3, p_1) + f(3, p_2) & (\text{for, } (t(t-1)) / 2 + 1 < d \leq m,\ t = 4) \\ 34 + 2f(3, p_1) + f(3, p_2) & (\text{for, } (t(t-1)) / 2 + 1 < d \leq m - 1,\ t = 5) \\ 58 + 2f(3, p_1) + f(3, p_2) & (\text{for, } d = m,\ t = 5) \end{cases}$$

III. Natural Algorithm (NA)

The core of natural algorithm is obtained from natural function 'f(t, d)'. According to this function, whole algorithm is classified into some parts for a specific number of towers and also a specific number of disks. Though there are some similarities in these but each of them has different procedures to perform the disk movement operations.

Let, 't' denotes the number of towers and 'd' denotes the number of disks in the puzzle and 'M' denotes the number of total movements.

1. If t = 3 then

a) Set, $M = 2^d - 1$.
b) Call f(3, d).

2. If t > d then
   a) Set, M = 2d – 1.
   b) Call f(t, d).
3. If $t \le d$ AND $d \le (t\ (t-1)) / 2$ then
   a) Set, M = 4d – 2t +1.
   b) Call f(t, d).
4. If $d = (t\ (t-1)) / 2 + 1$ then
   a) Set, $M = 2t^2 - 4t + 9$.
   b) Call f(t, d).
5. If $((t(t-1))/2+1) < d$ AND $d \le m$ AND t = 4 then
   a) If d mod 2 = 0 then
      i. Set, $M = 17 + 2^{((d - 2)/2 +1)}$
   b) If d mod 2 = 1 then
      i. Set, $M = 17 + 3*2^{(d - 3)/2}$
   c) Call f(4, d).
6. If $(t\ (t-1)) / 2 +1 < d$ AND $d \le m-1$ AND t = 5 then
   a) If d mod 2 = 0 then
      i. Set, $M = 31 + 2^{((d - 6)/2 +1)}$
   b) If d mod 2 = 1 then
      i. Set, $M = 31 + 3*2^{(d - 7)/2}$
   c) Call f(5, d).
7. If d = m AND t = 5 then
   a) If d mod 2 = 0 then
      i. Set, $M = 55 + 2^{((d - 16)/2 +1)}$
   b) If d mod 2 = 1 then
      i. Set, $M = 55 + 3*2^{(d - 17)/2}$
   c) Call f(5, m).

f(t, d) = 2^d – 1 when, t = 3

For any tower-disk combination, only 2 towers are involved in each movement. When the TOH puzzle is arranged with 3 towers, movements may involve the left most 2 towers or right most 2 towers or the left most and right most tower only. This involvement of the towers can be represented using 3 bit binary numbers as 110, 101 and 011. The left most bit denotes the left most tower, right most bit denotes the right most tower and the middle bit denotes the middle tower respectively. Bit 1 denotes the involvement of the tower and 0 denotes the absence of the tower. The mentioned binary numbers can also be written in decimal format as 6, 5, 3. Depending on the even or odd number of disks we get 2 sequences as shown below:

a) 6 then 5 then 3
b) 5 then 6 then 3.

Now let, each of the disk has a weight according to their size.(i.e. the smallest disk has weight 1 and the largest disk has the weight m), when there is no disk in a tower then it has a weight 0. 'w_l' and 'w_r' are the left and right tower's top most disk weight respectively. And the 2 patterns are cyclic in nature i.e. pattern a) 6, 5, 3, 6, 5, 3, 6, 5, 3, ….. and pattern b) 5, 6, 3, 5, 6, 3, 5, 6, 3,…. .

Thus the following steps are involved in a 3 towers m disks puzzle.

1. Check that, whether the total number of disks is even or odd.
2. If d mod 2 = 0 then
   a) Follow pattern a) 6 then 5 then 3.
3. If d mod 2 = 1 then
   a) Follow pattern b) 5 then 6 then 3.
4. If $w_l > w_r$ then
   a) Move the disk from right tower to left tower.
5. If $w_l < w_r$ then
   a) Move the disk from left tower to right tower.
6. Change the pattern, set the next stage of it.
7. Go back to step-4 to continue the process up to move all the disks to the destination tower.

f(t, d) = 2d – 1 when, t > d

When the puzzle is arranged using of multiple towers, where the number of towers are more than the number of disks i.e. t > d, then the observation is found that each of the disk takes 2 movements except the largest disk that takes only 1 movement. Using this concept the following algorithm is obtained considering the smallest disk as d_1, then the immediate large disk as d_2 and so on.

1. Move disk d_1 from t_l to t_{r-1} (if exists).
2. Move disk d_2 from t_l to t_{r-2} (if exists).
3. Continue the above movement process (in step-1 and step-2) up to move disks d_{m-1} from t_l to its appropriate position (if exists).
4. Move the largest disk d_m from t_l to t_r.
5. Now move disk d_{m-1} from its source tower to t_r (if exists).
6. Move disk d_{m-2} from its source tower to t_r (if exists).
7. Continue the above movement process (in step-5 and step-6) up to move all the disks to the destination tower t_r (if exists).

f(t, d) = 4d – 2t +1 when, $t \le d \le (t\ (t-1)) / 2$

In limit $t \le d \le (t\ (t-1)) / 2$ some disks of the puzzle take 4 movements, some take two and the largest disk takes only a single movement.

Now let t_m is the number of temporary towers. Thus the following steps are involved,

1. Move the top most disk from t_l to t_r.
2. Move the next disk from t_l to t_{r-1}.
3. Continue the above process (step-1 and step-2) up to fill all the temporary towers.
4. Decrement t_m by 1.
5. At t_l there are total $(d - t_m + 1)$ disks left. Now compare this value with t_m.
6. If $(d - t_m + 1) > t_m$ then
   a) Move back all the disks from t_{l+2} to t_r at t_{l+1} to make a disk stack.
   b) Continue the process from step-1 until the above condition is dis-satisfied.
7. If $(d - t_m + 1) \le t_m$ then
   a) Make a disk stack at $(t-d)^{th}$ tower position using the disks from $(t-d + 1)^{th}$ tower to t_r.
8. Move the top most disk at t_l to t_{r-1} (if exists), move the next disk at t_{r-2} (if exists). And continue the process up to move all the disks on the top of the largest disk d_m (if exist).
9. Move disk d_m from t_l to t_r.
10. Now search the previous small disk at bottom position from t_{r-1} to t_l. When found,
    a) Check is there any disk(s) on the top of this disk.

b) If exist(s) move the disk(s) to the left most empty tower (one by one).
c) Move the desired disk to t_r.

11. Continue the above movement process (step-9 and step-10) up to move all the disks at t_r.

$f(t, d) = 2t^2 - 4t + 9$ when, $d = (t\ (t-1)) / 2 + 1$

When, $d = (t\ (t-1)) / 2 + 1$ the nature of the puzzle is changed suddenly. Here the puzzle does not have any part of the above mentioned. At and after this limit the puzzle is divided in 2 or more sub puzzles. Using 4 towers it is divided into 2 sub puzzles. Using 5 towers it is divided into 3 and for other towers it is divided into more sub puzzles. The main theme of this limit is to make small disk stacks up to obtain a 3 towers 4 disks puzzle which is obtained actually at the last option.

In this case, the following steps are involved,

1. Move the disks (t-1) at t_{l+1}, (t-2) at t_{l+2}.
2. Continue the similar process to move the other disks until t be 3.
3. Move 4 disks using 3 towers 4 disks algorithm.
4. Move the smallest stack from t_{r-2} to t_r, then the stack from t_{r-3} to t_r.
5. Continue the similar process up to move all the disks at t_r from the temporary towers.

$f(t, d) = 20 + 2f(3, p_1) + f(3, p_2)$ when, $(t\ (t-1)) / 2 + 1 < d \leq m$, $t = 4$

Above the limit $(t\ (t-1)) / 2 + 1$, the function f(4, 3) is called 4 times and f(3, p) is called 3 times only. Here $f(3, p) = 2*f(3, p_1) + f(3, p_2)$, where $p = p_1 + p_2$. The value of P_1 and p_2 are obtained from an even-odd operation.

So, the following steps are involved in this part of the algorithm,

1. Use the 4 towers 3 disks algorithm to move 3 disks from t_l to t_{l+1}.
2. Check the number of disks is even or odd.
3. If odd then
   a) Call 3 towers (d / 2) disks algorithm to move the disks at t_{l+2}.
4. If even then
   a) Call 3 towers (d / 2) (floor value) disks algorithm to move the disks at t_{l+2}.
5. Recall 4 towers 3 disks algorithm to move the disks from t_{l+1} to t_{l+2}.
6. For odd number of disks recall 3 towers (d / 2) disks algorithm or, for even number of disks call 3 towers (d / 2) (ceiling value) disks algorithm to move the rest disks from t_l to t_r.
7. Recall 4 towers 3 disks algorithm at t_{l+2} to move the disks at t_{l+1} back.
8. Recall the used algorithm at step-3 or step-4 to move the disks from t_{l+2} to t_r.
9. Again call 4 towers 3 disks algorithm at t_{l+1} to move the disks at t_r.

$f(t, d) = 34 + 2f(3, p1) + f(3, p2)$ when, $(t\ (t-1)) / 2 + 1 < d \leq m-1$, $t = 5$

Above the limit $(t\ (t-1)) / 2 + 1 < d$ both the 4 towers and 5 towers puzzle themes are almost similar. But still 5 towers puzzle has some dissimilarity. And that's why; it is classified in some more parts. Here, the following steps are involved,

1. Use 5 towers 4 disks algorithm to move 4 disks from t_l to t_{l+1}.
2. Use 4 towers 3 disks algorithm to move 3 disks from t_l to t_{l+2}.
3. Check the number of disks is even or odd.
4. If odd then
   a) Call 3 towers (d / 2) disks algorithm to move the disks at t_{l+3}.
5. If even then
   a) Call 3 towers (d / 2) (floor value) disks algorithm to move the disks at t_{l+3}.
6. Recall 4 towers 3 disks algorithm to move the disks from t_{l+2} to t_{l+3}.
7. For odd number of disks recall 3 towers (d / 2) disks algorithm or, for even number of disks call 3 towers (d / 2) (ceiling value) disks algorithm to move the rest disks from t_l to t_r.
8. Recall 4 towers 3 disks algorithm to move the disks from t_{l+3} to t_{l+2}.
9. Recall the used algorithm at step-3 or step-4 to move the disks from t_{l+3} to t_r.
10. Recall 4 towers 3 disks algorithm to move the disks from t_{l+2} to t_r.
11. Recall 5 towers 4 disks algorithm to move the disks from t_{l+1} to t_r.

$f(t, d) = 58 + 2f(3, p1) + f(3, p2)$ when, $d = m$, $t = 5$

This function is the extension of the function $f(t, d) = 34 + 2f(3, p_1) + f(3, p_2)$ in between the limit $(t\ (t-1)) / 2 + 1 < d \leq m-1$. Thus following steps are found in the movements of the disks.

1. Use 5 towers 4 disks algorithm to move 4 disks from t_l to t_{l+1}.
2. Use 4 towers 3 disks algorithm to move 3 disks from t_l to t_{l+2}.
3. Check the number of disks is even or odd.
4. If odd then
   a) Call 3 towers (d / 2) disks algorithm to move the disks at t_{l+3}.
5. If even then
   a) Call 3 towers (d / 2) (floor value) disks algorithm to move the disks at t_{l+3}.
6. Recall 4 towers 3 disks algorithm to move the disks from t_{l+2} to t_{l+3}.
7. Recall 5 towers 4 disks algorithm to move the disks from t_{l+1} to t_{l+3}.
8. For odd number of disks recall 3 towers (d / 2) disks algorithm or, for even number of disks call 3 towers (d / 2) (ceiling value) disks algorithm to move the rest disks from t_l to t_r.
9. Recall 5 towers 4 disks algorithm to move the disks from t_{l+3} to t_{l+1}.
10. Recall 4 towers 3 disks algorithm to move the disks from t_{l+3} to t_{l+2}.
11. Recall the used algorithm at step-3 or step-4 to move the disks from t_{l+3} to t_r.
12. Recall 4 towers 3 disks algorithm to move the disks from t_{l+2} to t_r.
13. Recall 5 towers 4 disks algorithm to move the disks from t_{l+1} to t_r.

IV. N-Towers Solution

The nature of towers of Hanoi puzzle depends on the number of disks provided in the puzzle for a specific number of towers. For this reason, for a specific number of disks; after a few towers the movements of the disks is to be constant. Then the increment of towers does not have any effects on the movements of the disks. This constant combination is found at t > d. Thus, t = d + 1. After the value of this 't', the extra towers do not have any effects. As in this paper, maximum 15 disks are used, so we have, t = 15 + 1 = 16. Thus the 16 towers 15 disk solution is the 'n' towers solution. This is shown in the following table (Table-1).

Table 1: Number of movements of n towers for 15 disks.

t / d	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	3	3	3	3	3	3	3	3	3	3	3	3	3	3
3	7	5	5	5	5	5	5	5	5	5	5	5	5	5
4	15	9	7	7	7	7	7	7	7	7	7	7	7	7
5	31	13	11	9	9	9	9	9	9	9	9	9	9	9
6	63	17	15	13	11	11	11	11	11	11	11	11	11	11
7	127	25	19	17	15	13	13	13	13	13	13	13	13	13
8	255	33	23	21	19	17	15	15	15	15	15	15	15	15
9	511	41	27	25	23	21	19	17	17	17	17	17	17	17
10	1023	49	31	29	27	25	23	21	19	19	19	19	19	19
11	2047	65	39	33	31	29	27	25	23	21	21	21	21	21
12	4095	81	47	37	35	33	31	29	27	25	23	23	23	23
13	8191	113	55	41	39	37	35	33	31	29	27	25	25	25
14	16383	145	63	45	43	41	39	37	35	33	31	29	27	27
15	32767	209	71	49	47	45	43	41	39	37	35	33	31	29

V. Conclusion

Towers of Hanoi puzzle is not just a puzzle but also a popular mathematical problem that needs special tools for its solution. When the number of towers are increased by keeping the number of disks same, it takes simple mathematical and logical terms that are totally different from any combination of 3 towers and multiple disks. Again keeping the number of towers fixed, when the number of disks are increased, it creates the problem that is too critical. In this paper n-towers solution is given for 15 numbers of disks but NA provides the scope to enhance the number of disks and towers as per requirement. The NA developed here is simple in the sense that it gives an easy solution to a large number of tower & disks combinations.

References

[1] http://en.wikipedia.org/wiki/Tower_of_Hanoi
[2] http://www.cut-the-knot.org/recurrence/hanoi.shtml
[3] http://online-judge.uva.es/problemset/v104/10444.html
[4] http://mazeworks.com/hanoi/index.htm
[5] http://www.superkids.com/aweb/tools/logic/towers
[6] Gwenny T.L. Janssen at al., "Celeration of Executive Functioning while Solving the Tower of Hanoi: Two Single Case Studies Using Protocol Analysis", International Journal of Psychology and Psychological Therapy, 2010, 10, 1, pp. 19-40.
[7] Janssen G, De Mey H, & Egger J (2009). Executive functioning in college students: Evaluation of the Dutch Executive Function Index (EFI-NL). International Journal of Neuroscience, 119, 792-805.
[8] Koorland MA & MacLeod S (2005). The accuracy improvement measure: A tool for assessing the effectiveness of teacher preparation programs. Journal of Precision Teaching and Celeration, 21, 13-18.
[9] Egger J, De Mey H, & Janssen G (2007). Assessment of executive functioning in psychiatric disorders: functional diagnosis as the ouverture of treatment. Clinical Neuropsychiatry, 4, 111-116.
[10] Egger J, De Mey H, & Janssen G (2007). Assessment of executive functioning in psychiatric disorders: functional diagnosis as the ouverture of treatment. Clinical Neuropsychiatry, 4, 111-116.
[11] Cicerone K, Levin HS, Malec J, Stuss D, & Whyte J (2006). Cognitive rehabilitation interventions for executive function: Moving from bench to bedside in patients with traumatic brain injury. Journal of Cognitive Neuroscience, 18, 1212-1222.

A. S. Zaforullah Momtaz has been serving as Teaching Assistant of the Department of Computer Science and Engineering of University of Asia Pacific. He has completed his Bachelor degree in Computer Science and Engineering from The University of Asia Pacific, in the year 2010.

Md. Fayyaz Khan has been serving as an Associate professor of the Department of Electrical and electronics engineering of United International University. He was the former head of Electrical and electronics engineering and Computer Science and Engineering department of Ahsanullah University of science and technology and The University of Asia pacific respectively. He was also a former faculty of Islamic University of technology, University of Bahrain and Bangladesh university of Engineering and Technology (BUET).

Muhammad Sajjad Hossain has been serving as a Lecturer in Mathematics with the Department of Mathematics Ahsanullah University of Science and Technology (AUST), Dhaka, Bangladesh. He joined at AUST in September 2010. Before joining AUST he was a teacher of University of Asia Pacific, Dhaka, Bangladesh for more than 02 (two) years. Part-time faculty of Premier University, Chittagong, Bangladesh, AM of Azim Group, Chittagong and also S.O. of Agrani Bank, Artilari Branch, Bangladesh. He also earned three Gold Medals and UGC merit scholarship. He has already completed his MBA from UAP. Now he is continuing his M. Phil in BUET

Aloke Kumar Saha is Head of Computer Science and Engineering Department of University of Asia Pacific (UAP), Dhaka, Bangladesh. He usually teaches courses on Digital Logic Design, Numerical Methods, Data Structures, Discrete Mathematics, Computer Graphics and Basic Electrical Engineering. His current research interests are Algorithm, Artificial Intelligence and Software Development. For more than 13 (Thirteen) years, he is also working with the undergraduate students of UAP, as a part of their paper works, on the software development and implementation, Bi-Directional Heuristic Search Algorithm etc.

Kazi Shamsul Arefin has been serving as a Lecturer and Convener of Research and Publication Unit (RPU) with the Department of Computer Science and Engineering (CSE), University of Asia Pacific (UAP). He joined at UAP in October 2009 after completion his M.Sc. Engg. (CSE). Right now he teaches courses on Database Management System (DBMS), Software Development, and Programming on Artificial Intelligence and Expert Systems. Besides this, he is an Editor of Journal of Computer and Information Technology (IJCIT), ISSN 2078-5828 (Print) and 2218-5224 (Online),URL: www.ijcit.org. In addition to these, he is engaged in research activities throughout his undergraduate years and has 18 (Eighteen) research papers (in 3rd IEEE ICCSIT, ICCIT, ICECC, ICEESD, IJECS, IJCTE, IJCEE, IJCIT and so on). Moreover, he is a member of International Association of Computer Science and Information Technology (IACSIT), Membership No: 80337708. For more information please visit: www.arefin.comxa.com