Recursive/Non Recursive Algorithms to Generate Triangular-Rectangular Numbers Directly/Indirectly Using Pell's Equation through Python 3

Article *in* International Journal of Multi Discipline Science (IJ-MDS) · March 2019

Mehul Arun Jadhav
The New College Kolhapur

Sanjay Pandurang Thorat
Vivekanand College, Kolhapur

Peer Reviewed Referred and
UGC Listed Journal
(Journal No. 40776)

ISSN 2277 - 5730

AN INTERNATIONAL MULTIDISCIPLINARY QUARTERLY RESEARCH JOURNAL

Volume-VIII, Issue-I
January - March - 2019
English Part - VII /
Marathi Part - III

IMPACT FACTOR / INDEXING
2018 - 5.5
www.sjifactor.com

AJANTA

Ajanta Prakashan

ISSN 2277 - 5730
AN INTERNATIONAL MULTIDISCIPLINARY QUARTERLY RESEARCH JOURNAL

AJANTA

Volume - VIII Issue - I January - March - 2019

English Part - VII / Marathi Part - III

Peer Reviewed Refereed and UGC Listed Journal

Journal No. 40776

ज्ञान-विज्ञान विमुक्तये

IMPACT FACTOR / INDEXING
2018 - 5.5
www.sjifactor.com

❖ EDITOR ❖
Asst. Prof. Vinay Shankarrao Hatole
M.Sc (Maths), M.B.A. (Mktg.), M.B.A. (H.R.),
M.Drama (Acting), M.Drama (Prod. & Dir.), M.Ed.

❖ PUBLISHED BY ❖

Ajanta Prakashan
Aurangabad. (M.S.)

EDITORIAL BOARD

VOLUME - VIII, ISSUE - I - JANUARY - MARCH - 2019
AJANTA - ISSN 2277 - 5730 - IMPACT FACTOR - 5.5 (www.sjifactor.com)

CONTENTS OF ENGLISH PART - VII

8. Recursive/Non Recursive Algorithms to Generate Triangular - Rectangular Numbers Directly / Indirectly Using Pell's Equation through Python 3

Mr. Jadhav Mehul Arun
Assistant Professor, Department of Computer Science (Entire), The New College, Kolhapur, Maharashtra, Shivaji University, Kolhapur.
Mr. Patankar Sanjay Pandurang
Associate Professor, Head Department of Mathematics, Vivekanand College, Kolhapur, Maharashtra, Shivaji University, Kolhapur.
Mr. Thorat Sanjay Pandurang
Associate Professor, Head Department of Mathematics, Lal Bahadur Shashtri College of Arts, Science and Commerce, Satara, Maharashtra, Shivaji University, Kolhapur.
Mr. Kulkarni Harshawardhan Raghunath
Research Student, Department of Computer Science (Entire), The New College, Kolhapur, Maharashtra, Shivaji University, Kolhapur.

Abstract

The aim of this research paper is to study the properties of Triangular-Rectangular numbers with the help of recurrence relations and its solution which will lead to minimize the time complexity and space complexity for such algorithms using Pell's equation.

Keywords: TR Number, Triangular number, Rectangular number, Pell's Equation, Pell-Fermat Equation, Python 3, Recurrence Relation, Chi-Squared Test.

1. Introduction

Definition 1:(Triangular Numbers) Non negative integers represented by $t_n = \frac{n(n+1)}{2}$ are called as triangular numbers. e.g. 0,1,3,6, ...

Definition 2:(Rectangular Numbers) Non negative integers represented by $t_n = n(n+1)$ are called as rectangular numbers i.e. product of two consecutive Natural numbers. e.g. 0,2,6,12, ...

Definition 3:(Triangular-Rectangular Numbers/TR numbers) Non negative integers which are simultaneously triangular as well rectangular are called Triangular-Rectangular Numbers. e.g. 0,6,210,7140, ...

2. Recurrence Relation of TR number, solution, ratio at infinity & Source Code in Python 3:

Result 1: TR Numbers are generated using recurrence relation

$$a_n = 34a_{n-1} - a_{n-2} + 6, a_0 = 0, a_1 = 6$$

and its solution is

$a_n = \frac{1}{32}\left((3+2\sqrt{2})(17+12\sqrt{2})^n + (3-2\sqrt{2})(17-12\sqrt{2})^n - 6\right)$ hence this problem is NP(Non Polynomial time)

Result 2: At infinity ratio between two TR numbers is $(17+12\sqrt{2})$ and we can convert NP to P(Polynomial time)

Proof:

$\lim_{n\to\infty} \frac{a_{n+1}}{a_n} = (17+12\sqrt{2})$ hence $a_n = \left[(17+12\sqrt{2})a_{n-1} + 6\right], a_0 = 0$ which is again NP.

O_i	E_i	$\frac{(O_i - E_i)^2}{E_i}$
6	6	
209.8233765	210	0.000148552
7139.818177	7140	$4.63019E-06$
242555.818	242556	$1.36526E-07$
8239769.818	8239770	$4.01915E-09$
279909629.8	279909630	$1.18313E-10$
9508687656	9508687656	$3.48279E-12$
$3.23015E+11$	$3.23015E+11$	$1.02485E-13$
$1.0973E+13$	$1.0973E+13$	$3.00677E-15$
Total		0.000153323

Table 1: Chi Squared Test

As $\chi^2_{7,1} = 0.008151 > \frac{(O_i-E_i)^2}{E_i}$, hence error decreases.

Recurrence Relation 1 $a_n = 34a_{n-1} - a_{n-2} + 6,$ $a_0 = 0, a_1 = 6$	import math
	def TR1(n) :
	  if n==0 :
	    return 0
	  if n==1 :
	    return 6
	  return (TR1(n-1)*34-TR1(n-2)+6)

Solution 1	import math def TR2(n) : return ((((3+2*math.sqrt(2))*(17+12*math.sqrt(2))**n)+((3-

	2*math.sqrt(2))*(17-12*math.sqrt(2))**n)-6)/32)

Recurrence Relation 2	import math
	def TR3(n) :
	  if n==0 :
	    return 0
	  return math.ceil((17+12*math.sqrt(2))*TR3(n-1)+6)

Solution 2	import math def TR4(n) : return math.ceil((((17+12*math.sqrt(2))**n+1)*6/(1-(17+12*math.sqrt(2))))
Table 2 : Source Codes using Python 3	

3. Pell's Diophantine equation and Analysis using Recurrence Relation for Variables

Definition 4: (Pell's Equation/ Pell–Fermat equation)

Pell's equation is any Diophantine equation of the form where is a given positive non-square integer and integer solutions are sought for and . In Cartesian coordinates, the equation has the form of a hyperbola; solutions occur wherever the curve passes through a point whose x and y coordinates are both integers, such as the trivial solution with and . Joseph Louis Lagrange proved that, as long as is not a perfect square, Pell's equation has infinitely many distinct integer solutions.

Result 3: If —— represents triangular number and represents rectangular number then Negative Pell's equation for it is,

i.e. are always odd and as is not square it has infinitely many solutions in .

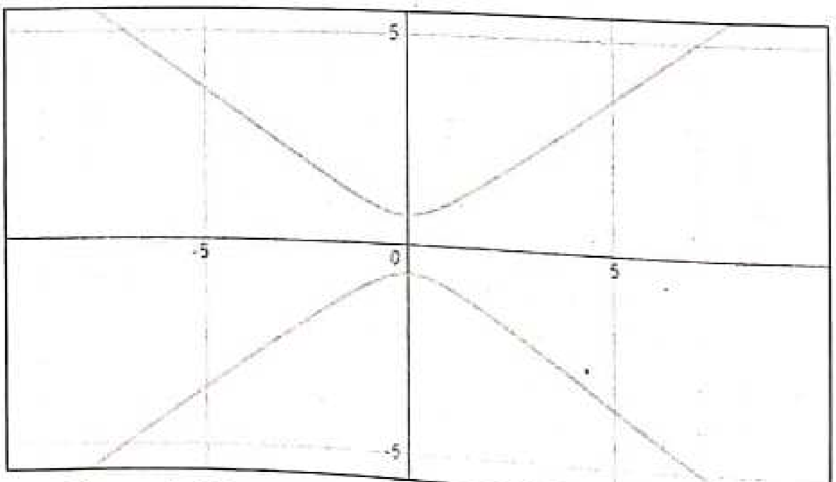

Figure 1: Hyperbola for Negative Pell's Equation

Result 4: All TR numbers are at even term i.e. in $t_m = m(m+1), m = 2k, k \epsilon Z^+$

Proof:

As $\frac{n(n+1)}{2} = m(m+1)$

I) For Odd:

By putting $n = 2k_{n,o} + 1, k_{n,o} \epsilon Z^+$ i.e. odd number we get $m(m+1) = (k_1 + 1)(2k_{n,o} + 1)$

As $(k_{n,o} + 1) < (2k_{n,o} + 1), k_{n,o} \epsilon Z^+$ we get m is even and $k_{n,o}$ is odd.

II) For Even:

By putting $n = 2k_{n,e}, k_{n,e} \epsilon Z^+$ i.e. even number we get $m(m+1) = k_{n,e}(2k_{n,e} + 1)$

As $k_{n,e} < (2k_{n,e} + 1), k_{n,e} \epsilon Z^+$ we get m is even and $k_{n,e}$ is also even.

X	Y	N	M	$K_{n,e}$	$K_{n,o}$	K_m
1	1	0	0	0		0
7	5	3	2		1	1
41	29	20	14	10		7
239	169	119	84		59	42
1393	985	696	492	348		246
8119	5741	4059	2870		2029	1435
47321	33461	23660	16730	11830		8365
275807	195025	137903	97512		68951	48756
1607521	1136689	803760	568344	401880		284172

Table 3: All Variables

4. Analysis using Recurrence for X, Y, M, N, K and implementation using Python

We can generate above variables using formulae and source code in python 3 to minimize calculation and memory space.

Recurrence Relation 1 $a_n = 6a_{n-1} - a_{n-2}$, $a_0 = 1, a_1 = 7$	import math def X1(n) : if n==0 : return 1 if n==1 : return 7 return (X1(n-1)*6-X1(n-2))
Solution 1 $a_n = \left(\left(\frac{\sqrt{2}+1}{2\sqrt{2}}\right)(3+2\sqrt{2})^n + \left(\frac{\sqrt{2}-1}{2\sqrt{2}}\right)(3-2\sqrt{2})^n \right)$	import math def X2(n) : return ((math.sqrt(2)+1)/(2*sqrt(2))*(3+2*math.sqrt(2))**n)+((3-2*math.sqrt(2))**n*((math.sqrt(2)-1)/(2*sqrt(2))
Recurrence Relation 2	import math def X3(n) :

$a_n = \lceil(3+2\sqrt{2})a_{n-1}\rceil$, $a_1 = 7$	if n==1 : return 7 return math.ceil((3+2*math.sqrt(2))*X3(n-1))
Table 4 : X	

Recurrence Relation 1
$a_n = 6a_{n-1} - a_{n-2}$,
$a_0 = 1, a_1 = 5$
	import math
	def Y1(n) :
	  if n==0 :
	    return 1
	  if n==1 :
	    return 5
	  return (X1(n-1)*6-X1(n-2))

Solution 1 $a_n = \left(\left(\frac{\sqrt{2}+1}{2}\right)(3+2\sqrt{2})^n + \left(\frac{\sqrt{2}-1}{2}\right)(3-2\sqrt{2})^n\right)$	import math def Y2(n) : return ((math.sqrt(2)+1)/(2)*(3+2*math.sqrt(2))**n)+((3-2*math.sqrt(2))**n*((math.sqrt(2)-1)/(2))

Recurrence Relation 2
$a_n = \lfloor(3+2\sqrt{2})a_{n-1}\rfloor$,
$a_0 = 1$
	import math
	def X3(n) :
	  if n==0 :
	    return 1
	  return math.floor((3+2*math.sqrt(2))*X3(n-1))

Table 5 : Y	

Recurrence Relation 1
$a_n = 6a_{n-1} - a_{n-2} + 2$,
$a_0 = 0, a_1 = 2$
	import math
	def M1(n) :
	  if n==0 :
	    return 0
	  if n==1 :
	    return 2
	  return (X1(n-1)*6-X1(n-2)+2)

Solution 1 $a_n = \frac{1}{8}\left((2+\sqrt{2})(3+2\sqrt{2})^n + (2-\sqrt{2})(3-2\sqrt{2})^n\right)$	import math def M2(n) : return (2+(math.sqrt(2))/(8)*(3+2*math.sqrt(2))**n)+((3-2*math.sqrt(2))**n*((2-math.sqrt(2))/(8)

Recurrence Relation 2
$a_n = \lceil(3+2\sqrt{2})a_{n-1} + 2\rceil$,
$a_0 = 0$
	import math
	def M3(n) :
	  if n==0 :
	    return 0
	  return math.ceil((3+2*math.sqrt(2))*M3(n-1)+2)

Table 6 : M	

Recurrence Relation 1
$a_n = 6a_{n-1} - a_{n-2} + 2$,
$a_0 = 0, a_1 = 3$
	import math
	def N1(n) :
	  if n==0 :
	    return 0
	  if n==1 :

	return 3 return (N1(n-1)*6-N1(n-2)+2)
Solution 1 $a_n = \frac{1}{16}\left(\begin{matrix}(2+\sqrt{2})(3+2\sqrt{2})^n + \\ (2-\sqrt{2})(3-2\sqrt{2})^n\end{matrix}\right)$	import math def N2(n) : return ((math.sqrt(2)+1)/(2)*(3+2*math.sqrt(2))**n)+((3-2*math.sqrt(2))**n*((math.sqrt(2)-1)/(2))
Recurrence Relation 2 $a_n = \lfloor(3+2\sqrt{2})a_{n-1} + 3\rfloor,$ $a_0 = 0$	import math def X3(n) : if n==0 : return 0 return math.floor((3+2*math.sqrt(2))*X3(n-1)+3)
Table 7 : N	
Recurrence Relation 1 $a_n = 6a_{n-1} - a_{n-2} + 1,$ $a_0 = 0, a_1 = 1$	import math def KM1(n) : if n==0 : return 0 if n==1 : return 1 return (KM1(n-1)*6-KM1(n-2)+1)
Solution 1 $a_n = \frac{1}{16}\left(\begin{matrix}(2+\sqrt{2})(3+2\sqrt{2})^n + \\ (2-\sqrt{2})(3-2\sqrt{2})^n\end{matrix}\right)$	import math def KM2(n) : return (2+(math.sqrt(2))/(16)*(3+2*math.sqrt(2))**n)+((3-2*math.sqrt(2))**n*((2-math.sqrt(2))/(16)
Recurrence Relation 2 $a_n = \lceil(3+2\sqrt{2})a_{n-1} + 1\rceil,$ $a_0 = 0$	import math def KM3(n) : if n==0 : return 0 return math.ceil((3+2*math.sqrt(2))*KM3(n-1)+2)
Table 8 : K_m	
Recurrence Relation 1 $a_n = 34a_{n-1} - a_{n-2} + 8,$ $a_0 = 0, a_1 = 10$	import math def KME1(n) : if n==0 : return 0 if n==1 : return 10 return (KME1(n-1)*34-KME1(n-2)+8)
Solution 1 $a_n = \left(\begin{matrix}\left(\frac{7+5\sqrt{2}}{4}\right)(17+12\sqrt{2})^n + \\ \left(\frac{7-5\sqrt{2}}{4}\right)(17-12\sqrt{2})^n - \frac{3}{4}\end{matrix}\right)$	import math def KME 2(n) : return (((7+5*math.sqrt(2))/4*(17+12*math.sqrt(2))**n)+((7-5*math.sqrt(2))/4*(17-12*math.sqrt(2))**n)-3/4)
Recurrence Relation 2	import math

$a_n = \left[(17 + 12\sqrt{2})a_{n-1} + 25\right],$ $a_0 = 1$	def KME3(n) :
	  if n==0 :
	    return 1
	  return math.ceil((17+12*math.sqrt(2))*KME3(n-1)+25)

Table 9 : $K_{n,e}$	

Recurrence Relation 1 $a_n = 34a_{n-1} - a_{n-2} + 24,$ $a_0 = 1, a_1 = 59$	import math
	def KMO1(n) :
	  if n==0 :
	    return 1
	  if n==1 :
	    return 59
	  return (KMO1(n-1)*34- KMO1(n-2)+24)

Solution 1 $a_n = \left(\left(\frac{1+\sqrt{2}}{8}\right)(17+12\sqrt{2})^n + \left(\frac{1-\sqrt{2}}{8}\right)(17-12\sqrt{2})^n - \frac{1}{4}\right)$	import math def KMO2(n) : return (((1+math.sqrt(2))/8*(17+12*math.sqrt(2))**n)+((1-math.sqrt(2))/8*(17-12*math.sqrt(2))**n)-1/4)

Recurrence Relation 2 $a_n = \left[(17 + 12\sqrt{2})a_{n-1} + 9\right],$ $a_0 = 0$	import math
	def KMO3(n) :
	  if n==0 :
	    return 1
	  return math.ceil((17+12*math.sqrt(2))*KMO3(n-1)+9)

Table 10 : $K_{n,o}$	

5. Conclusion

It is clear that TR sequence can be generated using recursive as well as non recursive algorithm and complexity can be converted from NP to P using recurrence relation which minimised time as well as space complexity. Similarly we may apply this study to other recursive algorithms also.

References

1. Barbeau, Edward J. (2003), Pell's Equation, Problem Books in Mathematics, Springer-Verlag, ISBN 0-387-95529-1
2. Liu, Zhi-Guo (2003-12-01). "An Identity of Ramanujan and the Representation of Integers as Sums of Triangular Numbers". The Ramanujan Journal. 7 (4): 407–434. ISSN 1382-4090
3. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. Introduction to Algorithms, Second Edition. MIT Press and McGraw-Hill. 1990. ISBN 0-262-03293-7.