Recursive Definition and Recurrence Relations

- Recursive Definition
  - Sequences
  - Sets
  - Functions
  - Algorithms
- Recurrence Relations
- Solution of Recurrence Relations
  - Proving Closed Form with Induction
  - Substitution Method

Imdad ullah Khan

Recurrence Relations

Recurrence relation is an equation that recursively defines a sequence

Useful for modeling and solving many counting problems

- Fibonacci sequence
- Number of bacteria doubling every hour
- Number of moves required to solve the tower of Hanoi puzzle
- Number of operations performed by a (recursive) algorithm

Fibonacci Numbers

Problem:

A young pair of rabbits (one of each gender) on an island

- A pair of rabbits does not breed until they are 2 months old
- Each pair of rabbits produces another pair each month

▷ *Assumption:* No rabbits ever die

Find the number of pairs of rabbits in the island after n months

Originally studied by Leonardo Fibonacci, in his book 'Liber abaci' (The Book of Calculation) 13^{th} century

Fibonacci Numbers

Reproducing pairs (at least two months old)	Young pairs (less than two months old)	Month	Reproducing pairs	Young pairs	Total pairs
		1	0	1	1

Fibonacci Numbers

Reproducing pairs (at least two months old)	Young pairs (less than two months old)	Month	Reproducing pairs	Young pairs	Total pairs
		1	0	1	1
		2	0	1	1

Fibonacci Numbers

Reproducing pairs (at least two months old)	Young pairs (less than two months old)	Month	Reproducing pairs	Young pairs	Total pairs
		1	0	1	1
		2	0	1	1
		3	1	1	2

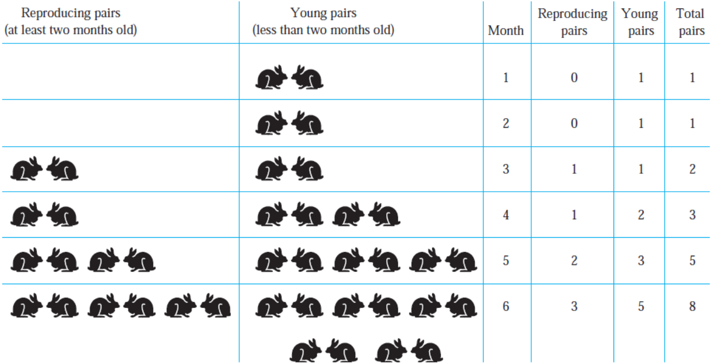
Fibonacci Numbers

Reproducing pairs (at least two months old)	Young pairs (less than two months old)	Month	Reproducing pairs	Young pairs	Total pairs
		1	0	1	1
		2	0	1	1
		3	1	1	2
		4	1	2	3

Fibonacci Numbers

Reproducing pairs (at least two months old)	Young pairs (less than two months old)	Month	Reproducing pairs	Young pairs	Total pairs
		1	0	1	1
		2	0	1	1
		3	1	1	2
		4	1	2	3
		5	2	3	5

Fibonacci Numbers



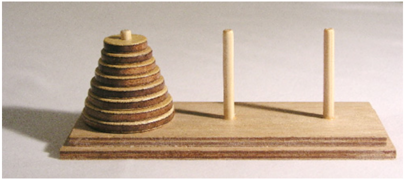
Reproducing pairs (at least two months old)	Young pairs (less than two months old)	Month	Reproducing pairs	Young pairs	Total pairs
		1	0	1	1
		2	0	1	1
		3	1	1	2
		4	1	2	3
		5	2	3	5
		6	3	5	8

Let f_n be number of pairs of rabbits in month n

- f_{n-1} : number of pairs in previous month
- f_{n-2} : number of new pairs in month n, why?
  - ▷ a new pair is from a pair ≥ 2 months old

$$f_n = \begin{cases} 1 & n = 0 \\ 1 & n = 1 \\ f_{n-1} + f_{n-2} & n \geq 2 \end{cases}$$

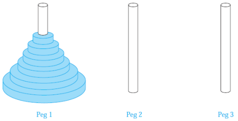
Tower of Hanoi Puzzle



- A popular puzzle invented by *Édouard Lucas* in 19^{th} century
- Three pegs mounted on a board together with disks of different sizes
- Initially all disks are placed on peg 1 in increasing order of size
- Move all disks to peg 3 in the same order
  - **Rule 1:** Can only move one disk at a time
  - **Rule 2:** Can never place a larger disk over a smaller one

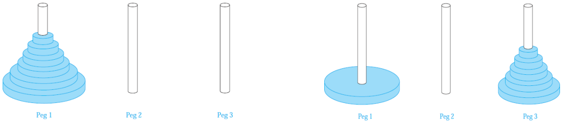
How many moves are required for n disks?

Solving Hanoi Tower Puzzle: Initial State



Initial Configuration

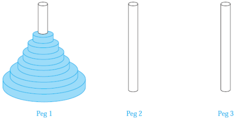
Solving Hanoi Tower Puzzle: Initial State



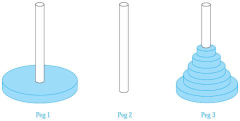
Initial Configuration **Step 1**

1. Transfer the top $n-1$ disks from Peg 1 to Peg 3 following rules

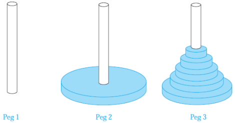
Solving Hanoi Tower Puzzle: Initial State



Initial Configuration



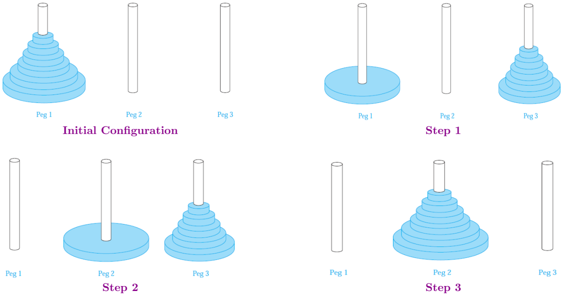
Step 1



Step 2

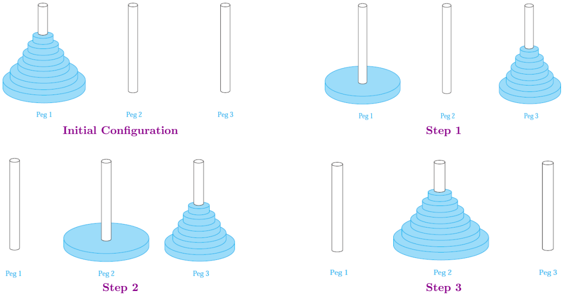
1. Transfer the top $n-1$ disks from Peg 1 to Peg 3 following rules
2. Transfer the largest disk to peg 2

Solving Hanoi Tower Puzzle: Initial State



1. Transfer the top $n-1$ disks from Peg 1 to Peg 3 following rules
2. Transfer the largest disk to peg 2
3. Transfer $n-1$ disks from peg 3 to peg 2 (place atop the largest disk)

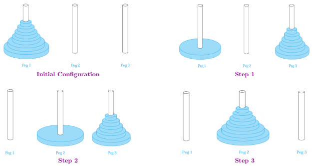
Solving Hanoi Tower Puzzle: Initial State



1. Transfer the top $n-1$ disks from Peg 1 to Peg 3 following rules
2. Transfer the largest disk to peg 2
3. Transfer $n-1$ disks from peg 3 to peg 2 (place atop the largest disk)

How many moves in total?

Solving Hanoi Tower Puzzle: Initial State



1. Transfer the top $n-1$ disks from Peg 1 to Peg 3 following rules
2. Transfer the largest disk to peg 2
3. Transfer $n-1$ disks from peg 3 to peg 2 (place atop the largest disk)

H_n : number of moves performed by the above procedure for n disks

ICP 13-1 Write a recurrence relation for H_n ?

- Step 1 takes H_{n-1} moves
- Step 2 takes 1 move
- Step 3 takes H_{n-1} moves

$$H_n = \begin{cases} 1 & \text{if } n = 1 \\ 2H_{n-1} + 1 & \text{if } n > 1 \end{cases}$$

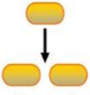
Bacteria Doubling Every Hour

Hour: 0  $\boxed{\text{1 cell}}$ $a_0 = 2^0$ *(initial)*

Bacteria Doubling Every Hour

Hour: 0

Hour: 1

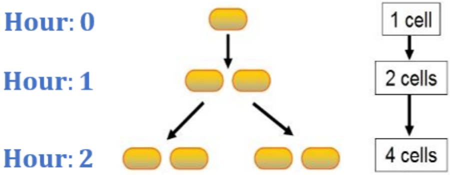


$\boxed{\text{1 cell}} \rightarrow \boxed{\text{2 cells}}$

$a_0 = 2^0$ *(initial)*

$a_1 = 2 \times a_o = 2^1$

Bacteria Doubling Every Hour

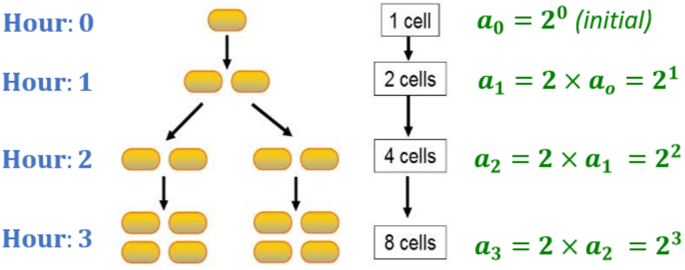


$a_0 = 2^0$ *(initial)*

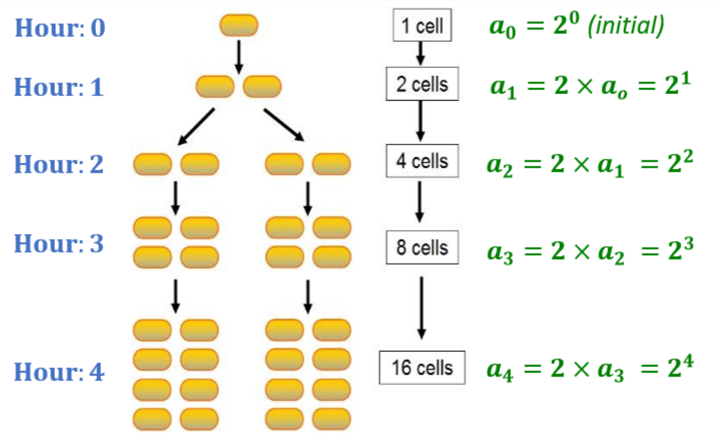
$a_1 = 2 \times a_o = 2^1$

$a_2 = 2 \times a_1 = 2^2$

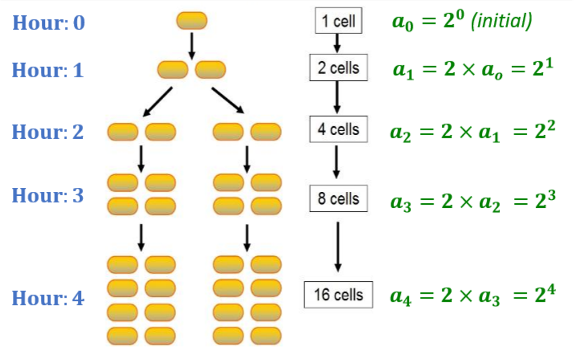
Bacteria Doubling Every Hour



Bacteria Doubling Every Hour



Bacteria Doubling Every Hour



Let a_n be the number of bacteria at hour n

ICP 13-2 Write a recurrence relation for a_n ?

$$a_n = \begin{cases} 1 & \text{if } n = 0 \\ 2 \times a_{n-1} & \text{if } n > 1 \end{cases}$$

Recursive Algorithms and Recurrences

Runtime analysis: Find the number of operations performed by algorithm

▷ This is a measure of the algorithm running time

Runtime of recursive algorithms are usually modeled by recurrences

▷ Closed form formulae for recursive functions (recurrences) proved by induction

Let $T(n)$ be number of comparisons performed by BIN-SEARCH on $|A| = n$

Each call to the function BIN-SEARCH makes

- some comparisons
- a recursive call

$$T(n) = \begin{cases} 1 & \text{if } n < 1 \\ T(n/2) + 3 & \text{if } n \geq 1 \end{cases}$$

BIN-SEARCH performs at most $2\log n$ comparisons i.e., $T(n) \leq 2\log n$

Properties of Recurrence Relations

A linear homogeneous recurrence relation of order k with constant coefficients is of the form

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \ldots + c_k a_{n-k}$$

- Linearity: earlier terms $a_{n-1}, \ldots, a_{n-k}$ appear as separate terms and to the first power
- Homogeneity: All terms have the same total degree (no constant term)
- Order: The expression for a_n contains the previous k(=order) terms
- Constant coefficient: $c_1, c_2, \ldots, c_k$ are constant (no dependency on n)

Properties of Recurrence Relations

A linear homogeneous recurrence relation of order k with constant coefficients is of the form

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \ldots + c_k a_{n-k}$$

Which of the following recurrences are linear homogeneous with constant coefficients. Write the order of those that are.

ICP 13-3 $f_n = f_{n-1}^2 + f_{n-2}$ ▷ not linear

ICP 13-4 $f_n = 3f_{n-1} - 2f_{n-2} + 7f_{n-3}$ ▷ linear, homogeneous of order 3

ICP 13-5 $f_n = 3f_{n-1} + 2f_{n-2} + 7f_{n-3} + 10$ ▷ not homogeneous

ICP 13-6 $f_n = nf_{n-1}$ ▷ not constant coefficient

ICP 13-7 $f_n = 2f_{n-2} + 5f_{n-7}$ ▷ linear, homogeneous of order 7

Sequence and its defining recurrence(s)

Consider the following recurrences and their properties

- $a_n = 1$ $\triangleright$ *(order 0)*
- $a_n = a_{n-1}$; $a_0 = 1$ $\triangleright$ *(order 1, homogeneous)*
- $a_n = 2a_{n-1} - 1$; $a_0 = 1$ $\triangleright$ *(order 1, non-homogeneous)*
- $a_n = 2a_{n-1} - a_{n-2}$; $a_0 = 1, a_1 = 1$ $\triangleright$ *(order 2, homogeneous)*

All these recurrences result in the same sequence:

$$1, 1, 1, 1, \cdots$$

A recursive sequence may not have a unique recurrence relation

Is the converse true? Is the sequence determined by a recurrence relation (initial terms + rule) unique?