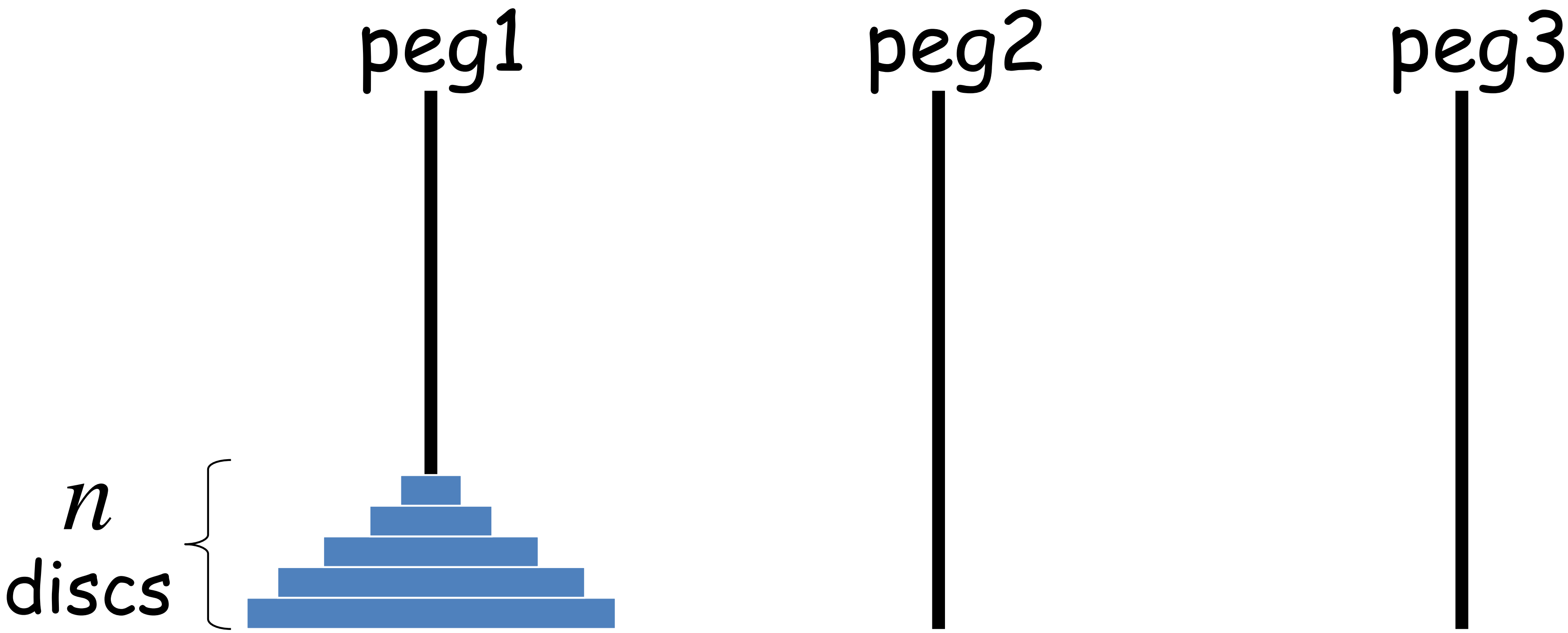
Discrete Mathematics

Advanced Counting Techniques II

Towers of Hanoi

Towers of Hanoi



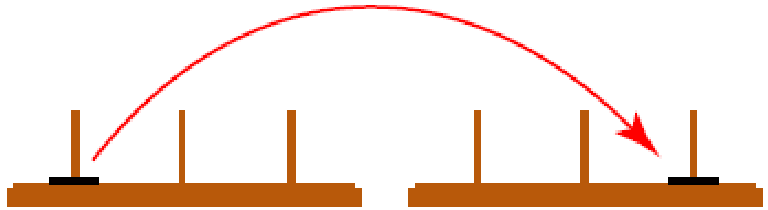
Goal: move all discs to bar3

Rule: not allowed to put larger discs on top of smaller discs

Counting Problem

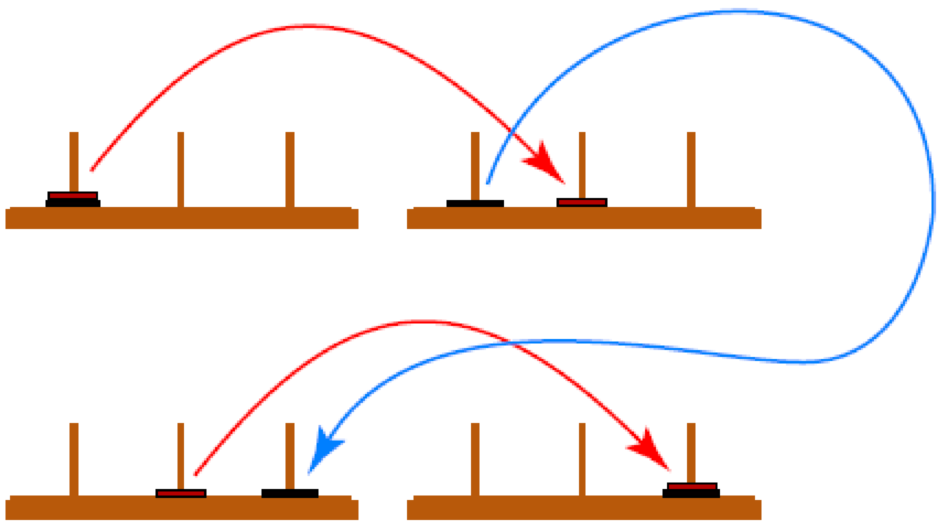
Count the minimum number of legal moves required to complete a tower of Hanoi puzzle that has n disks.

Tower of Hanoi: One Disk Solution



1 Move.

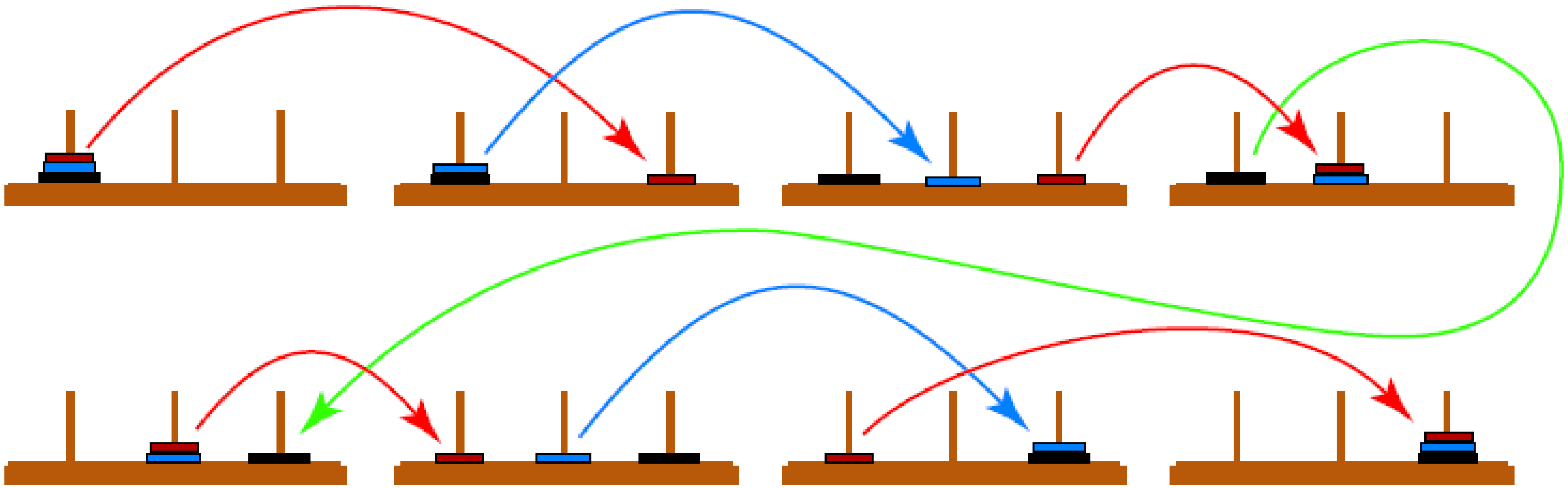
Two Disk Solution



Note that we perform the solution to the one-disk Tower of Hanoi *twice* (once in the top row, and once in the bottom row). Between rows, we move the bottom disk. Thus, we require

$$2 \cdot 1 + 1 = 3 \text{ Moves}$$

Tower of Hanoi: Three Disk Solution



Note that we perform the solution to the two-disk Tower of Hanoi *twice* (once in the top row, and once in the bottom row). Between rows we move the bottom disk. Thus, we require

$$2(2 \cdot 1 + 1) + 1 = 7 \text{ Moves}$$

Tower of Hanoi: What we know so far

Let a_n denote the minimum number of legal moves required to complete a tower of Hanoi puzzle that has n disks.

n	a_n
1	1
2	3
3	7

Tower of Hanoi: What we know so far

Let a_n denote the minimum number of legal moves required to complete a tower of Hanoi puzzle that has n disks.

n	a_n
1	1
2	3
3	7

Following the pattern, for $n = 4$ we need to solve the three-disk puzzle twice, plus one more operation to move the largest disk. Thus,

$$a_4 = 2a_3+1$$

Tower of Hanoi: What we know so far

Let a_n denote the minimum number of legal moves required to complete a tower of Hanoi puzzle that has n disks.

n	a_n
1	1
2	3
3	7

Following the pattern, for $n = 4$ we need to solve the three-disk puzzle twice, plus one more operation to move the largest disk. Thus,

$$a_4 = 2a_3+1$$

Similarly, for $n = 5$ disks, we expect that we will need to perform

$$a_5 = 2a_4+1$$

Tower of Hanoi: n Disk Analysis

Let a_n denote the minimum number of legal moves required to complete a tower of Hanoi puzzle that has n disks.

- Before the largest disk (i.e., the n-th disk) can be moved to the rightmost peg, all of the remaining ($n - 1$) disks must moved to the center peg. (These $n - 1$ disks must be somewhere, and they can't obstruct the transfer of the largest disk.) This requires a_{n-1} legal moves.

Tower of Hanoi: n Disk Analysis

Let a_n denote the minimum number of legal moves required to complete a tower of Hanoi puzzle that has n disks.

- Before the largest disk (i.e., the n-th disk) can be moved to the rightmost peg, all of the remaining (n -1) disks must moved to the center peg. (These n - 1 disks must be somewhere, and they can't obstruct the transfer of the largest disk.) This requires a_{n-1} legal moves.
- It takes 1 more operation to move the n-th disk to the rightmost peg.

Tower of Hanoi: n Disk Analysis

Let a_n denote the minimum number of legal moves required to complete a tower of Hanoi puzzle that has n disks.

- Before the largest disk (i.e., the n-th disk) can be moved to the rightmost peg, all of the remaining (n -1) disks must moved to the center peg. (These $n - 1$ disks must be somewhere, and they can't obstruct the transfer of the largest disk.) This requires a_{n-1} legal moves.
- It takes 1 more operation to move the n-th disk to the rightmost peg.
- Finally, another legal sequence of a_{n-1} steps is required to move the $n - 1$ disks from the center peg, to the rightmost peg.

Tower of Hanoi: n Disk Analysis

Let a_n denote the minimum number of legal moves required to complete a tower of Hanoi puzzle that has n disks.

- Before the largest disk (i.e., the n-th disk) can be moved to the rightmost peg, all of the remaining ($n - 1$) disks must moved to the center peg. (These $n - 1$ disks must be somewhere, and they can't obstruct the transfer of the largest disk.) This requires a_{n-1} legal moves.
- It takes 1 more operation to move the n-th disk to the rightmost peg.
- Finally, another legal sequence of a_{n-1} steps is required to move the $n - 1$ disks from the center peg, to the rightmost peg.

We thus obtain the recurrence relation,

$$a_n = 2a_{n-1}+1$$

Tower of Hanoi: Solution

With the solution for a single disk

$$a_1 = 1$$

the recurrence relation

$$a_n = 2a_{n-1}+1$$

Tower of Hanoi: Solution (Iterative Method)

$$a_n = 2a_{n-1} + 1 \qquad a_1 = 1$$

$$\begin{aligned}
a_n &= 2a_{n-1} + 1 \\
&= 2(2a_{n-2} + 1) + 1 = 2^2 a_{n-2} + 2 + 1 \\
&= 2^2(2a_{n-3} + 1) + 2 + 1 = 2^3 a_{n-3} + 2^2 + 2 + 1 \\
&\vdots \\
&= 2^{n\text{-}1} a_1 + 2^{n-2} + \cdots + 2 + 1 \\
&= 2^{n\text{-}1} + 2^{n-2} + \cdots + 2 + 1 \\
&= 2^n - 1
\end{aligned}$$

$$\sum_{i=0}^{n} 2^i = 2^{n+1} - 1$$

Prove: $a_n = 2^n - 1$ by induction:

1. Show the base case is true: $a_1 = 2^1 - 1 = 1$
2. Now assume true for a_k
3. Show true for a_{k+1}

$$
\begin{aligned}
a_{k+1} &= 2a_k + 1 \\
&= 2(2^k - 1) + 1 \\
&= 2^{k+1} - 1
\end{aligned}
$$

Tower of Hanoi: Solution

With the solution for a single disk

$$a_1 = 1$$

the recurrence relation

$$a_n = 2a_{n-1}+1$$

defines the solution

$$a_n = 2^n - 1$$

this algorithm would be $O(2^n)$

Tower of Hanoi: Solution

With the solution for a single disk

$$a_1 = 1$$

the recurrence relation

$$a_n = 2a_{n-1}+1$$

This is a Linear **<u>NonHomogenous</u>** Recurrence Relations with constant coefficients.

Tower of Hanoi: Solution

Solve this problem methodically. Rewrite:

$$a_n - 2a_{n-1} = 1$$

1) Solve with the RHS set to 0, i.e. solve the homogeneous case.
2) Add a particular solution to get general solution.

$$\boxed{\begin{array}{c}\text{General}\\ \text{Nonhomogeneous}\end{array}} = \boxed{\begin{array}{c}\text{General}\\ \text{homogeneous}\end{array}} + \boxed{\begin{array}{c}\text{Particular}\\ \text{Nonhomogeneous}\end{array}}$$

Tower of Hanoi: Solution

$$a_n - 2a_{n-1} = 1$$

1) Solve with the RHS set to 0, i.e. solve

$$a_n - 2a_{n-1} = 0$$

Characteristic equation: $r - 2 = 0$

so unique root is $r = 2$. General solution to homogeneous equation is

$$a_n^{(h)} = A \cdot 2^n$$

Tower of Hanoi: Solution

2) Add a particular solution to get general solution for $a_n - 2a_{n-1} = 1$. Use rule:

General Nonhomogeneous	=	General homogeneous	+	Particular Nonhomogeneous

There are little tricks for guessing particular nonhomogeneous solutions. For example, when the RHS is constant, the guess should also be a constant.

By Theorem 2(**PPT 26**) , a particular solution is of the form C: $a_n^{(p)} = C$.

Plug into the original relation:

$1 = C - 2C = -C$. Therefore $C = -1$.

General solution: $a_n = a_n^{(h)} + a_n^{(p)} = A \cdot 2^n + (-1) = A \cdot 2^n - 1$.

Tower of Hanoi: Solution

Finally, use initial conditions to get closed solution. In the case of the Towers of Hanoi recursion, initial condition is:

$$a_1 = 1$$

Using general solution $a_n = A \cdot 2^n - 1$ we get:

$1 = a_1 = A \cdot 2^1 - 1 = 2A - 1.$

Therefore, $2 = 2A$, so $A = 1$.

Final answer: $a_n = 2^n - 1$

Linear Nonhomogenous Recurrence Relations

Linear Nonhomogenous recurrence relation

A **Linear Nonhomogenous recurrence relation with constant coefficients**, that is a recurrence relation of the form

$a_n = c_1a_{n-1} + c_2a_{n-2} + \ldots + c_ka_{n-k} + F(n)$.

Where

- $c_1, c_2, \ldots, c_k$ are real numbers
- $F(n)$ is a function not identically zero depending only n
- $c_1a_{n-1} + c_2a_{n-2} + \ldots + c_ka_{n-k}$ is called the **associated homogenous recurrence relation**

Theorem 1: If the sequence $\{a_n^{(p)}\}$ is a particular solution of the recurrence relation $a_n = c_1 a_{n-1} + c_2 a_{n-2} + \ldots + c_k a_{n-k} + F(n)$ then every solution is of the form $\{a_n^{(p)} + a_n^{(h)}\}$, where $\{a_n^{(h)}\}$ is a solution of the associated homogeneous recurrence relation $a_n = c_1 a_{n-1} + c_2 a_{n-2} + \ldots + c_k a_{n-k}$.

Theorem 2: Suppose $\{a_n\}$ satisfies the nonhomogeneous recurrence relation with constant coefficients

$a_n=c_1a_{n-1}+c_2a_{n-2}+\ldots+c_ka_{n-k}+F(n)$,

and $F(n)=(b_tn^t+b_{t-1}n^{t-1}+\ldots+b_1n+b_0)s^n$.

When **s is not a root of the associate recurrence relation**, there is a particular solution ($a_n^{(p)}$)of the form

$(p_tn^t+p_{t-1}n^{t-1}+\ldots+p_1n+p_0)s^n$.

When **s is a root of the characteristic equation and its multiplicity is m**, there is a particular solution ($a_n^{(p)}$) of the form

$n^m(p_tn^t+p_{t-1}n^{t-1}+\ldots+p_1n+p_0)s^n$.

Solve the recurrence relation a_n-4a_{n-1} +3a_{n-2}=-200 where a_0=3000 and a_1=3300.

Step 1: Solve associated linear homogeneous by setting the RHS to 0, i.e. solve

$$a_n - 4a_{n-1} + 3a_{n-2} = 0$$

Characteristic equation: $r^2 - 4r + 3 = 0$

so roots are $r = 3, 1$. General solution to homogeneous equation is

$a_n^{(h)} = c_1(3^n) + c_2(1^n)$

$a_n-4a_{n-1}+3a_{n-2}=-200$ where $a_0=3000$ and $a_1=3300$

Step 2: Solve nonlinear part.

From theorem 2, we have $F(n) = (b_0)s^n$ where $s = 1$.

Since s = 1 and s is a root of the characteristic equation.

$a_n^{(p)} = n^1(b_0)s^n = n(b_0) = b_0 n$

We know that $a_n^{(p)}-4a_{n-1}^{(p)}+3a_{n-2}^{(p)}=-200$

$b_0 n-4b_0(n-1)+3b_0(n-2) = -200$

$b_0 n-4b_0 n+4b_0+3b_0 n-6b_0 = -200$

$-2b_0 = -200$

$b_0 = 100$

Then, $a_n^{(p)} = 100n$

$a_n - 4a_{n-1} + 3a_{n-2} = -200$ where $a_0 = 3000$ and $a_1 = 3300$

Step 3: Combine solution and solve for c_1 and c_2

$a_n^{(h)} = c_1(3^n)+c_2(1^n)$

$a_n^{(p)} = 100n$

$a_n = a_n^{(h)} + a_n^{(p)} = c_1(3^n)+c_2(1^n)+100n = c_1(3^n)+c_2+100n$

$a_0 = 3000$: We have $3000 = c_1+c_2$ ……………(1)

$a_1 = 3300$: We have $3300 = 3c_1+ c_2+100$ …..(2)

From (1), $c_2 = 3000 - c_1$

Substitute c_2 in 2, we have $3300 = 3c_1+ 3000-c_1+100$.

Then $c_1 = 100$ and $c_2 = 2900$.

Therefore,

$a_n = 100(3^n)+2900+100n$

Find the general solution to recurrence relation : $a_n = 2a_{n-1} + 2^{n-3} - a_{n-3}$

1) Rewrite as $a_n - 2a_{n-1} + a_{n-3} = 2^{n-3}$ and solve homogeneous part:

Characteristic equation: $r^3 - 2r + 1 = 0$.

Guess root $r = \pm 1$ as integer roots.

$r = 1$ works, so divide out by $(r - 1)$ to get

$r^3 - 2r + 1 = (r - 1)(r^2 + r - 1)$.

$r^3 - 2r + 1 = (r - 1)(r^2 + r - 1)$.

Quadratic formula on $r^2 + r - 1$:

$$r = (-1 \pm \sqrt{5})/2$$

So $r_1 = 1$, $r_2 = (-1+\sqrt{5})/2$, $r_3 = (-1-\sqrt{5})/2$

General homogeneous solution:

$$a_n^{(h)} = A + B\,[(-1+\sqrt{5})/2]^n + C\,[(-1-\sqrt{5})/2]^n$$

2) Nonhomogeneous particular solution to

$$a_n - 2a_{n-1} + a_{n-3} = 2^{n-3}$$

By Theorem 2, the particular solution ($a_n^{(p)}$) is of the form $k\,2^n$. Plug the solution in:

$$k\,2^n - 2k\,2^{n-1} + k\,2^{n-3} = 2^{n-3}$$

Coefficient Matching: $k = 1$.

So particular solution ($a_n^{(p)}$) is 2^n

General Nonhomogeneous	=	General homogeneous	+	Particular Nonhomogeneous

Final answer:

$$a_n = A + B\,[(-1+\sqrt{5})/2]^n + C\,[(-1-\sqrt{5})/2]^n + 2^n$$

Solving Recurrence Relations with Generating Function

Generating function:

$$G(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \cdots + a_k x^k + \cdots$$

$$= \sum_{k=0}^{\infty} a_k x^k$$

generating function for sequence

$$a_0, a_1, a_2, a_3, \ldots, a_k, \ldots$$

Can we find a_k with generating function?

Generating functions can also be used to solve recurrence relations

Example:

Solve recurrence relation

$$a_k = 3a_{k-1} \qquad a_0 = 2$$

$a_k = 3a_{k-1}$ $\quad$ $a_0 = 2$

Let $G(x)$ be the generating function for sequence $a_0, a_1, a_2, a_3, \ldots, a_k, \ldots$

$$G(x) = \sum_{k=0}^{\infty} a_k x^k$$

$$x \cdot G(x) = x\sum_{k=0}^{\infty} a_k x^k$$

$$= \sum_{k=0}^{\infty} a_k x^{k+1}$$

$$= \sum_{k=1}^{\infty} a_{k-1} x^k$$

xG(x) has a_{k-1} as the coefficient on a_k

G(x)= a_0+ a_1x+ a_2x^2+ a_3x^3+…+ a_kx^k+…

xG(x)= **0** +a_0x+ a_1x^2+ a_2x^3+…+ a_{k-1}x^k+…

$$a_k = 3a_{k-1} \qquad a_0 = 2$$

$$
\begin{aligned}
G(x) - 3xG(x) &= \sum_{k=0}^{\infty} a_k x^k - 3\sum_{k=1}^{\infty} a_{k-1} x^k \\
&= a_0 + \sum_{k=1}^{\infty} a_k x^k - \sum_{k=1}^{\infty} 3a_{k-1} x^k \\
&= a_0 + \sum_{k=1}^{\infty} (a_k x^k - 3a_{k-1} x^k) \\
&= a_0 + \sum_{k=1}^{\infty} \underbrace{(a_k - 3a_{k-1})}_{0} x^k \\
&= a_0 \\
&= 2
\end{aligned}
$$

$$G(x) - 3xG(x) = 2$$

$$\Downarrow$$

$$G(x) = \frac{2}{1 - 3x}$$

$$G(x) = 2\frac{1}{1-3x}$$

$$\sum_{k=0}^{\infty} \lambda^{x} k^{x} = \frac{1}{1-\lambda x}$$

$$G(x) = 2\sum_{k=0}^{\infty} 3^{k} x^{k}$$

$$G(x) = \sum_{k=0}^{\infty} 2 \cdot 3^{k} x^{k}$$

$$a_k = 3a_{k-1} \qquad a_0 = 2$$

$$\left.\begin{aligned} G(x) &= \sum_{k=0}^{\infty} 2 \cdot 3^k x^k \\ G(x) &= \sum_{k=0}^{\infty} a_k x^k \end{aligned}\right\} \Rightarrow$$

Solution to recurrence relation

$$a_k = 2 \cdot 3^k$$

Divide-and-Conquer Recurrence Relations

Divide-and-Conquer Recurrence Relations

- Suppose that
  - A recursive algorithm divides a problem of size n into a subproblems, where each subproblem is of size n/b.
  - A total of $g(n)$ extra operations are required in the conquer step of the algorithm to combine the solutions of the subproblems into a solution of the original problem.

- Then, if $f(n)$ represents the number of operations required to solve the problem of size n, it follows that f satisfies the recurrence relation

  - $f(n) = a f(n/b) + g(n)$

- This is called a **divide-and-conquer recurrence relation.**

Binary Search

- What happen in Binary search:
  - A subproblem is used: The algorithm reduces the search for an element in a search sequence of size n to the binary search for an element in a search sequence of size $n/2$, n even.
  - $g(n)$: two comparisons are needed to implement this reduction
    1. to determine which half of the list to use
    2. to determine whether any terms of the list remain

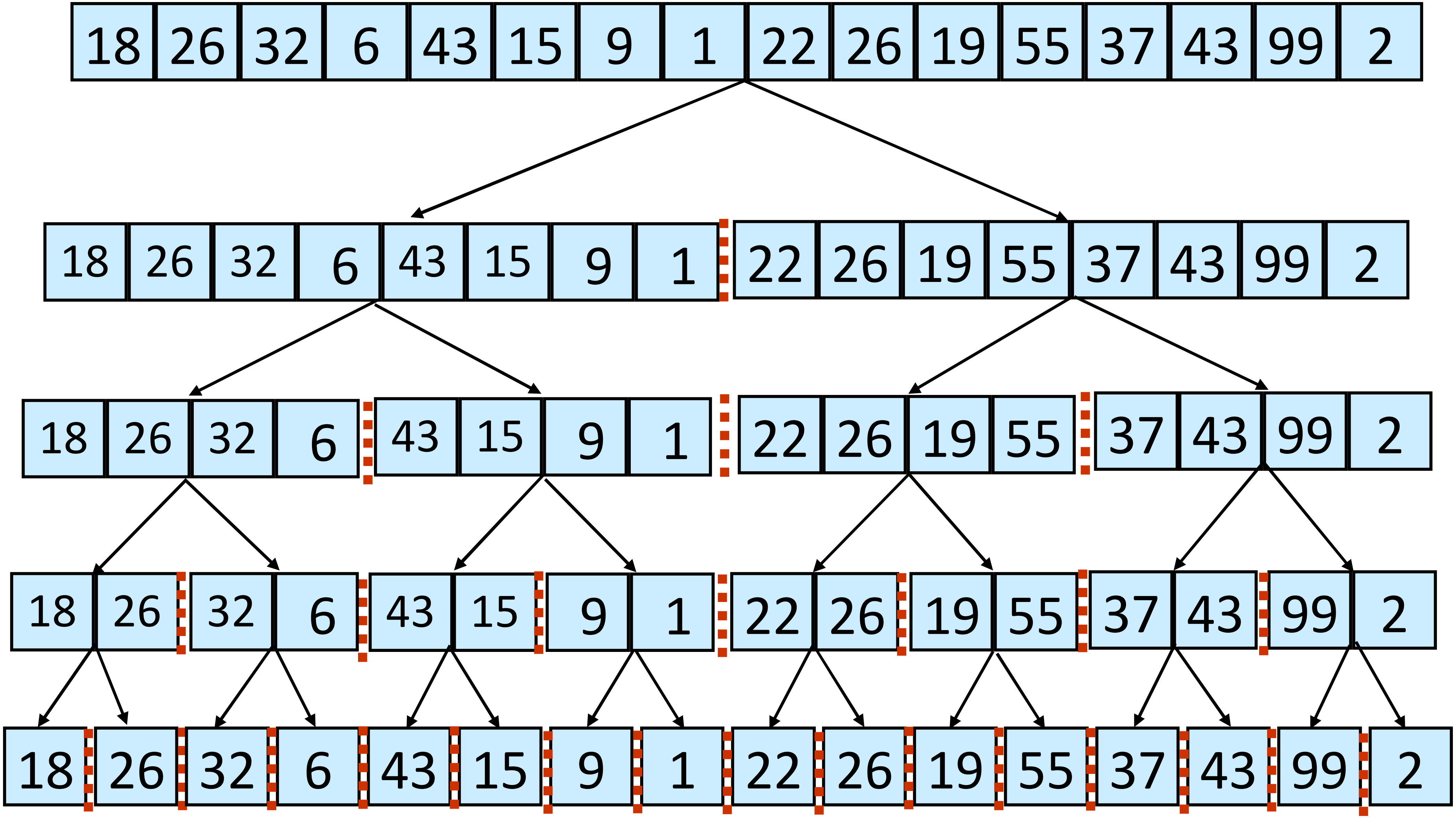
- If $f(n)$ represents the number of comparisons required to search for an element in a search sequence of size n, then
  - $\boldsymbol{f(n) = a\,f(n/b) + g(n) = f(n/2) + 2}$ **, $\boldsymbol{n}$ even**

Maximum and Minimum

- Locating the maximum and minimum elements of a sequence $a1, a2, \ldots, an$:
  - 2 subproblems are used:
    - If n = 1, then a1 is the maximum and minimum
    - If n > 1, then the sequence split into 2 sequences (either where both have the same number of elements or where one of the sequences has one ore element than the other), let say $n/2$
    - The problem is reduced to find the maximum and minimum of each of the two smaller sequences
    - The solution for original problem is the comparison result from these two smaller sequences
  - $g(n)$: two comparisons are needed to implement this reduction
    1. to compare the maxima of two sequences
    2. to compare the minima of two sequences

- If $f(n)$ represents the total number of comparisons needed to find then maximum and minimum elements of the sequence with n elements, then
  - $\boldsymbol{f(n) = a f(n/b) + g(n) = 2f(n/2) + 2}$ **, $\boldsymbol{n}$ even**

Merge Sort



Merge Sort

- What happen in Merge Sort:
  - Splits a list to be sorted with n items, where n even, into two lists with size $n/2$ elements each
  - $g(n)$: uses fewer than n comparisons to merge the sorted lists of $n/2$ items each into one sorted list

- Consequently, the number of comparisons used by the merge sort to sort a list of n element is less than $M(n)$ where,
  - $\boldsymbol{M(n) = a f(n/b) + g(n) = 2M(n/2) + n}$

Theorem 3

- Let $a, b \in \mathrm{N}$ and $c, d \in \mathrm{R}^{+}$ with $b > 1$.
  Let f be an increasing function that satisfies the recurrence relation $\boldsymbol{f(n) = a f(n/b) + c}$ and $f(1) = d$. Then

$$f(n) = \begin{cases} O\left(n^{\log_b a}\right) & \text{if } a > 1 \\ O(\log n) & \text{if } a = 1 \end{cases}$$

- Furthermore, when $n = b^k$, where k is a positive integer and $a > 1$;

$$f(n) = C_1 n^{\log_b a} + C_2 = C_1 a^k + C_2$$

where

$$C_1 = f(1) + c/(a-1) \quad \text{and} \quad C_2 = -c/(a-1)$$

Example

- Let $f(n) = 5f(n/2) + 3$ and $f(1) = 7$. Find $f\left(2^k\right)$ where k is a positive integer. Also estimate $f(n)$ if f is increasing function

Solution:

$a = 5$, $b = 2$, $c = 3$, then

$$
\begin{aligned}
f(n) &= a^k \left[f(1) + c/(a-1) \right] + \left[-c/(a-1) \right] \\
&= 5^k \left[7 + (3/4) \right] - 3/4 \\
&= 5^k (31/4) - 3/4
\end{aligned}
$$

If f is increasing function,

$$f(n) = O\left(n^{\log_b a}\right) = O\left(n^{\log_2 5}\right)$$

Binary Search

- Estimate the number of comparisons used by binary search

Solution:

When n is even,

$$f(n) = f(n/2) + 2$$

Where f is the number of comparisons required to perform a binary search on a sequence of size n. Hence,

$$f(n) = O(\log n)$$

Maximum and Minimum

- Estimate the number of comparisons used to locate the maximum and minimum elements.

Solution:

When n is even,

$$f(n) = 2f(n/2) + 2$$

Where f is the number of comparisons needed. Hence,

$$f(n) = O\left(n^{\log 2}\right) = O(n)$$

Theorem 4: Master Theorem

- Let a, b ϵ N and c, d ϵ R^+ with $b > 1$.
  Let f be an increasing function that satisfies the recurrence relation

$$f(n) = af(n/b) + cn^d$$

Then

$$f(n) = \begin{cases} O(n^d) & \text{if } a < b^d \\ O(n^d \log n) & \text{if } a = b^d \\ O(n^{\log_b a}) & \text{if } a > b^d \end{cases}$$

whenever $n = b^k$, where k is a positive integer

Merge Sort

- Estimate the number of comparisons used by the merge sort to sort a list of n elements.

Solution:

The number of comparisons is less than $M(n)$ where

$$M(n) = 2M(n/2) + n$$

Hence,

$$M(n) = O(n \log n)$$