Recursion

计划

递归是计算机科学中最重要的技术之一。

主要思想是捕获重复动作的不变量。

- 设置重复
  - 斐波那契重现
  - 解决问题的重复
  - 加泰罗尼亚语复发
- 解决重复问题

递归定义的序列

我们可以通过指定当前项和先前项之间的关系来定义一个序列。

- 算术序列:(a, a+d, a+2d, a+3d, ...,)递归定义:

  a0=a, ai+1=ai +d

- 几何序列:(a, ra, r2a, r3a, ...,) 递归定义:

  a0=a, ai+1=rai

- 谐波序列:(1, 1/2, 1/3, 1/4, ...,)

  递归定义: a0=1, ai+1=iai/(i+1)

兔子种群

兔子种群

- 每个月都会繁殖一对成熟的男孩/女孩兔子。
- 兔子在一个月后成熟。

wn ::= #第 n 个月的新生儿对

rn ::= #第 n 个月的配对

- 从一对新生儿开始： w0 =1, r0 = 0

n个月后有多少只兔子？

兔子种群

wn ::= #第 n 个月的新生儿对rn ::= #第n个月的

新生儿对

r1 = 1

rn = rn-1 + wn-1

wn = rn-1所以

rn = rn-1 + rn-2

兔子很容易过度繁殖,见兔子以澳大利亚为例。

研究兔子种群增长的是斐波那契。

我们将很快计算rn的封闭形式。

一个公式,使得计算rn的 # 步≤ 常数

暖身

我们将通过建立递归关系来解决计数问题。

首先,我们使用递归来计算我们已经知道的东西,以熟悉这种方法。

让我们计算 pow(S_n) 中的元素数量,其中S_n = {a_1 , a_2 , ..., a_n } 是一个 n 元素集。

设r_n为 pow(S_n) 的大小。

那么r_1 = 2,其中 pow(S_1) = {Φ, {a_1 }}

r_2 = 4,其中 pow(S_2) = {Φ, {a_1 }, {a_2 }, {a_1 ,a_2 }}

暖身

设rn是 pow(Sn) 的大小,其中Sn = {a1 , a2 , ..., an } 是一个 n 元素集。

那么r1 = 2,其中 pow(S1) = {Φ, {a1 }}

r2 = 4,其中 pow(S2) = {Φ, {a1 }, {a2 }, {a1 ,a2 }}

递归的主要思想是根据前面的ri定义rn 。

如何根据r1和r2 定义r3 ?

pow(S3) = {Φ, {a1 }, {a2 } 的并集, {a1 ,a2 }}

和 {a3 , {a1 ,a3 }, {a2 ,a3 }, {a1 ,a2 ,a3 }}

所以r3 = 2r2 。

而较低的集合是通过将a3添加到较高的集合而获得的。

暖身

设rn是 pow(Sn) 的大小,其中Sn = {a1 , a2 , ..., an } 是一个 n 元素集。

递归的主要思想是根据前面的ri定义rn 。

pow(Sn) = Sn-1 = {Φ, {a1 } 的并集， {a2 }, ..., {a1 ,a2 ,...,an-1 }}

和 {an , {a1 ,an }, {a2 ,an }, ..., {a1 ,a2 ,...,an-1 ,an }}

而较低的集合是通过在上面的集合中添加一个来获得的。

每个子集只计算一次。

所以rn = 2rn-1 ·

求解这个递归关系将显示rn = 2n （几何序列）。

没有特定模式的位串数

有多少个没有位模式 11 的 n 位字符串?

令rn为此类字符串的数量。

例如r1 = |{0, 1}| = 2,

r2 = |{00, 01, 10}| = 3

r3 = |{000, 001, 010, 100, 101}| = 5

r4 = |{0000, 0001, 0010, 0100, 0101, 1000, 1001, 1010}| = 8

你能看到图案吗?

r4 = |{0000, 0001, 0010, 0100, 0101} 联合

{1000, 1001, 1010}|

= 5 + 3 = 8

没有特定模式的位串数

有多少个没有位模式 11 的 n 位字符串?

令rn为此类字符串的数量。

我们如何使用r1 ,r2 ,...,rn-1 来计算它?

情况 1:第一位为 0。

然后可以将任何不带位模式 11 的 (n-1) 位字符串附加到末尾以

形成不带 11 的 n 位字符串。

所以在这种情况下,恰好有rn-1 个这样的 n 位字符串。

0 +

00000000000000000
00000000000000001
...
10101010101010101

没有 11 的所有 (n-1) 位字

符串的集合。

总共有rn-1个。

没有特定模式的位串数

有多少个没有位模式 11 的 n 位字符串?

令rn为此类字符串的数量。

我们如何使用r1 ,r2 ,...,rn-1 来计算它?

情况 2:第一位是 1。

那么第二位必须是 0,因为我们不能有 11。

然后可以将任何不带位模式 11 的 (n-2) 位字符串附加到末尾以

形成不带 11 的 n 位字符串。

所以在这种情况下,恰好有rn-2 个这样的 n 位字符串。

10+ _

0000000000000000
0000000000000001
...
1010101010101010

没有 11 的所有 (n-2) 位字符串的集合。

总共有rn-2个。

没有特定模式的位串数

有多少个没有位模式 11 的 n 位字符串?

0 +

0000000000000000
0000000000000001
...
101010101010101010101

没有 11 的所有 (n-1) 位字符串的集合。

总共有rn-1个。

10+ _

0000000000000000
0000000000000001
...
1010101010101010

没有 11 的所有 (n-2) 位字符串的集合。

总共有rn-2个。

没有位模式 11 的每个字符串都只计算一次。

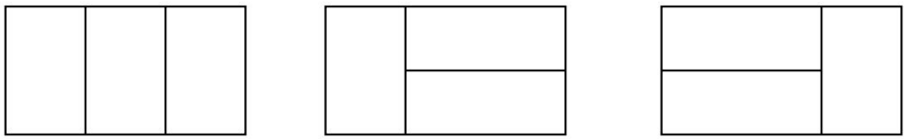
因此， rn = rn-1 + rn-2

锻炼

有多少个没有位模式 111 的 n 位字符串?

骨牌

给定一个 2xn 拼图,有多少种方法可以用多米诺骨牌（2x1 瓷砖)填充它?



例如,有 3 种方法可以用多米诺骨牌填充 2x3 拼图。

设rn是用多米诺骨牌填充 2xn 拼图的方法数。

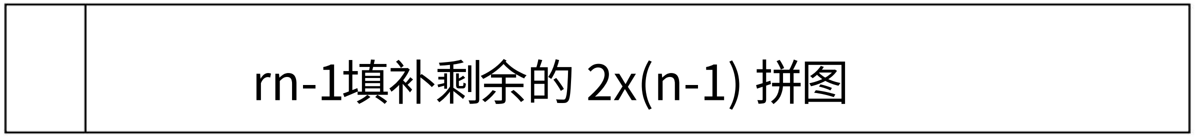
我们如何使用r1 ,r2 ,...,rn-1 来计算它?

骨牌

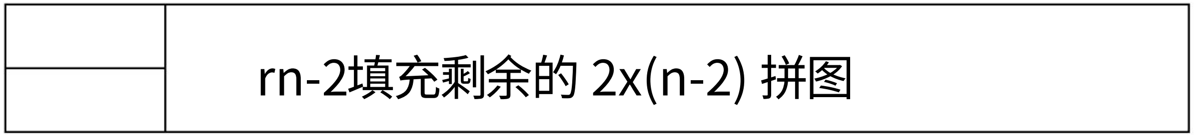
给定一个 2xn 拼图,有多少种方法可以用多米诺骨牌（2x1 瓷砖)填充它?

设rn是用多米诺骨牌填充 2xn 拼图的方法数。

案例1:垂直放置多米诺骨牌



案例2:水平放置多米诺骨牌

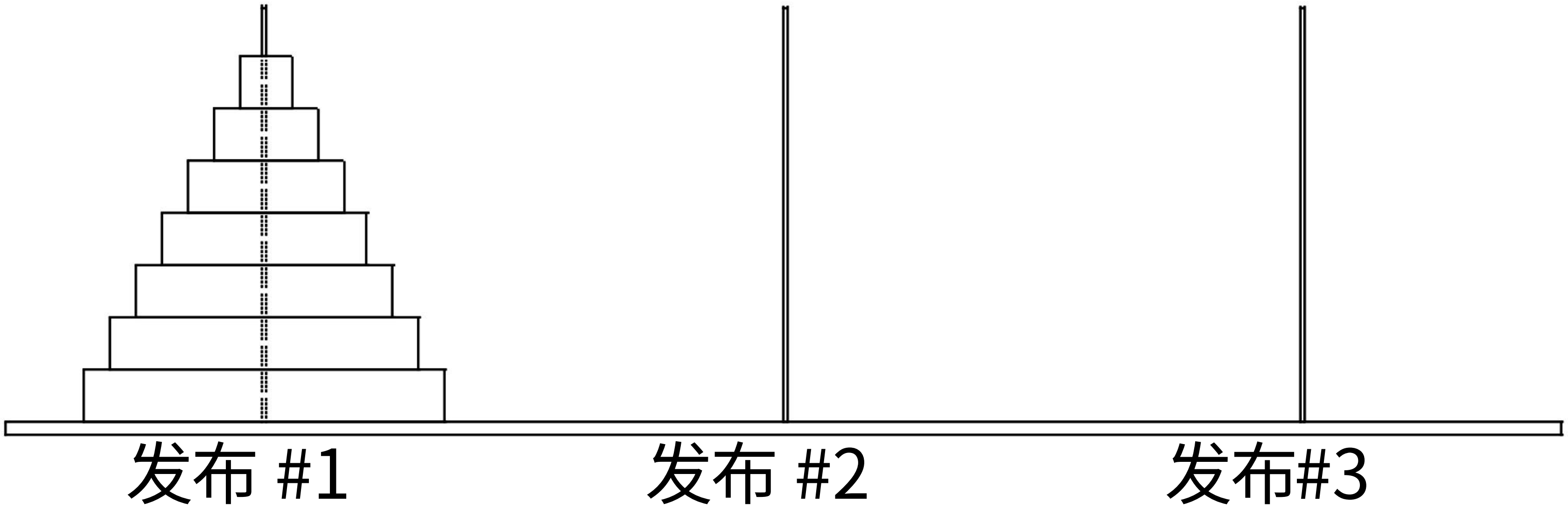


因此， rn = rn-1 + rn-2

计划

- 设置重复
  - 斐波那契重现
  - 解决问题的重复
  - 加泰罗尼亚语复发
- 解决重复问题

河内塔

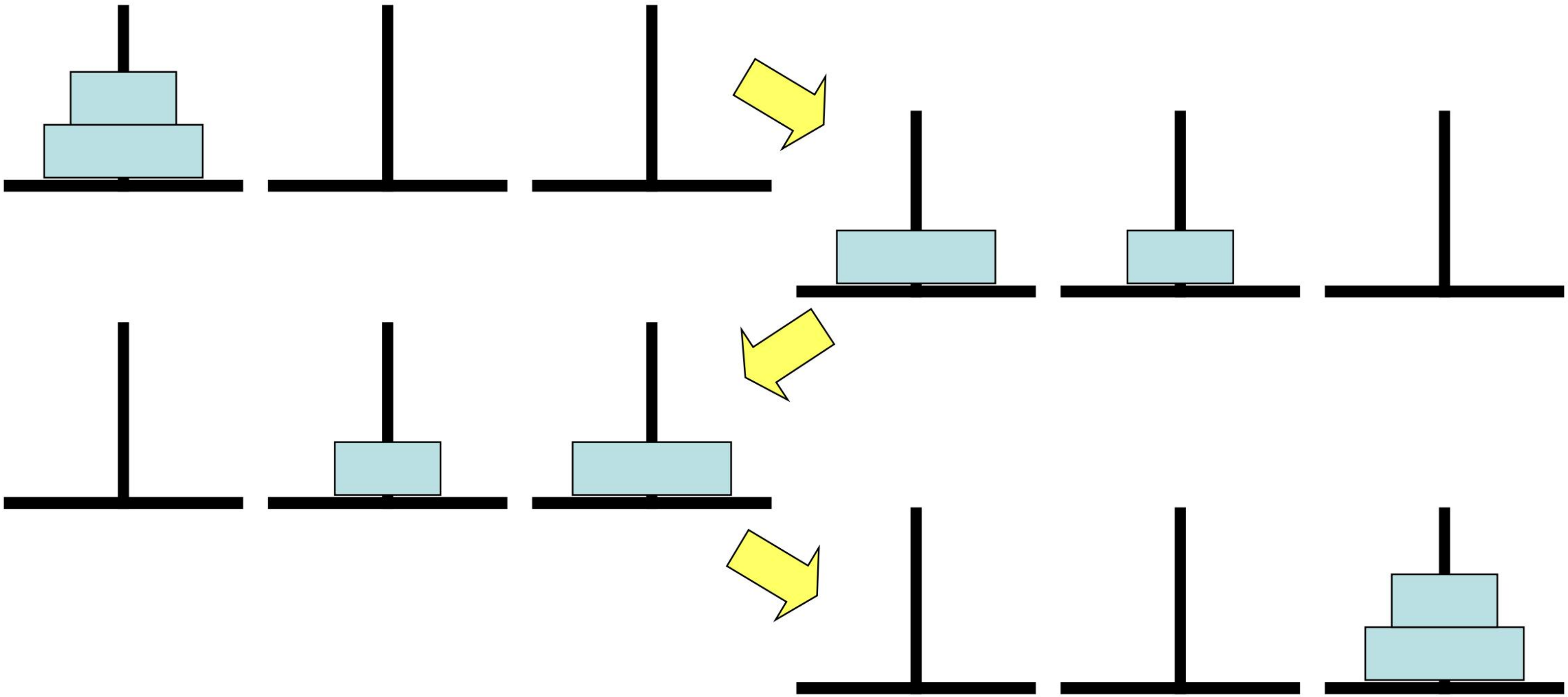


目标是将所有磁盘移动到后 3。

规则是较大的磁盘不能放在较小的磁盘上。

河内塔

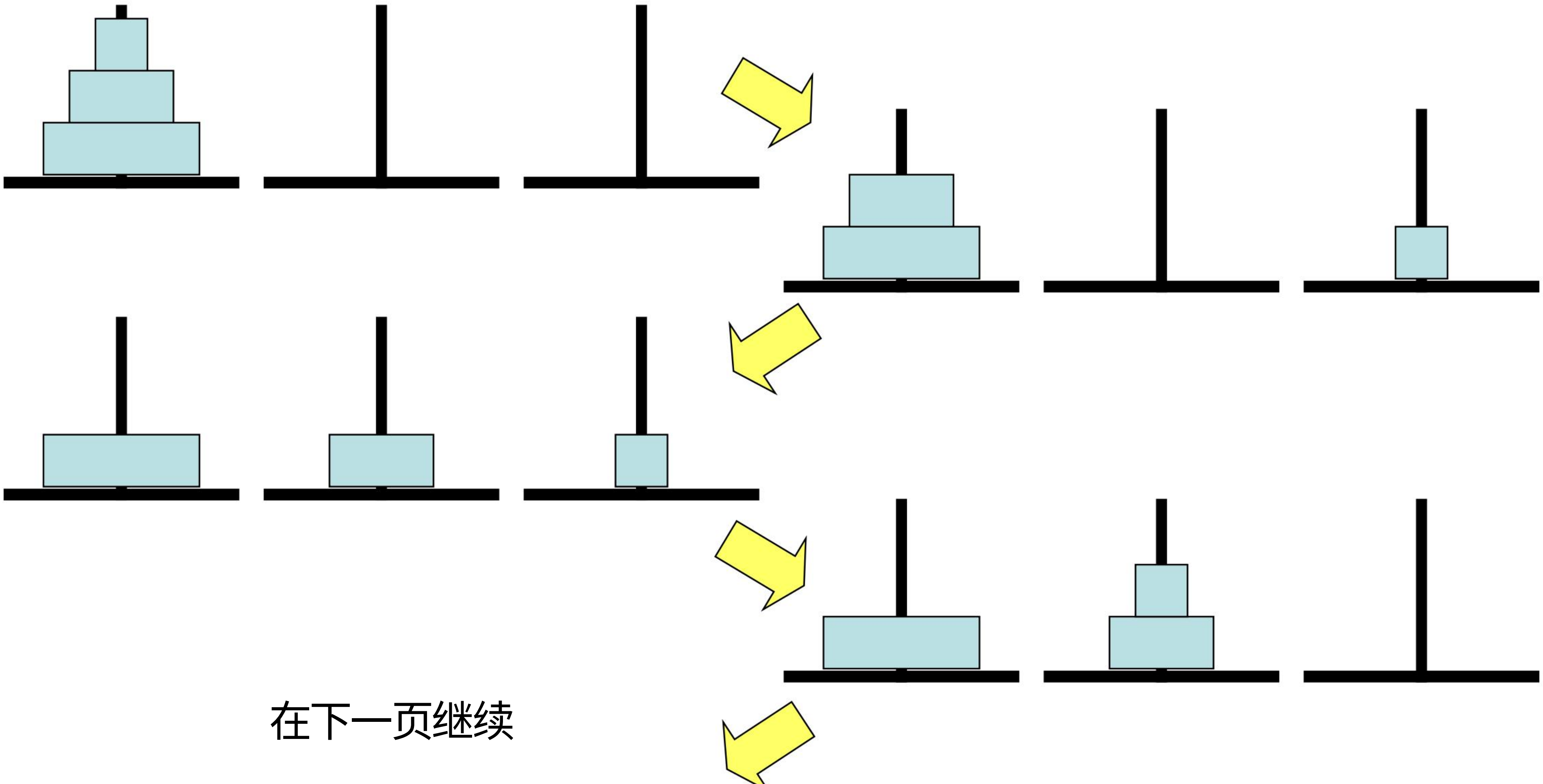
如果我们只有 2 个磁盘,这很容易。



一共3步。

河内塔

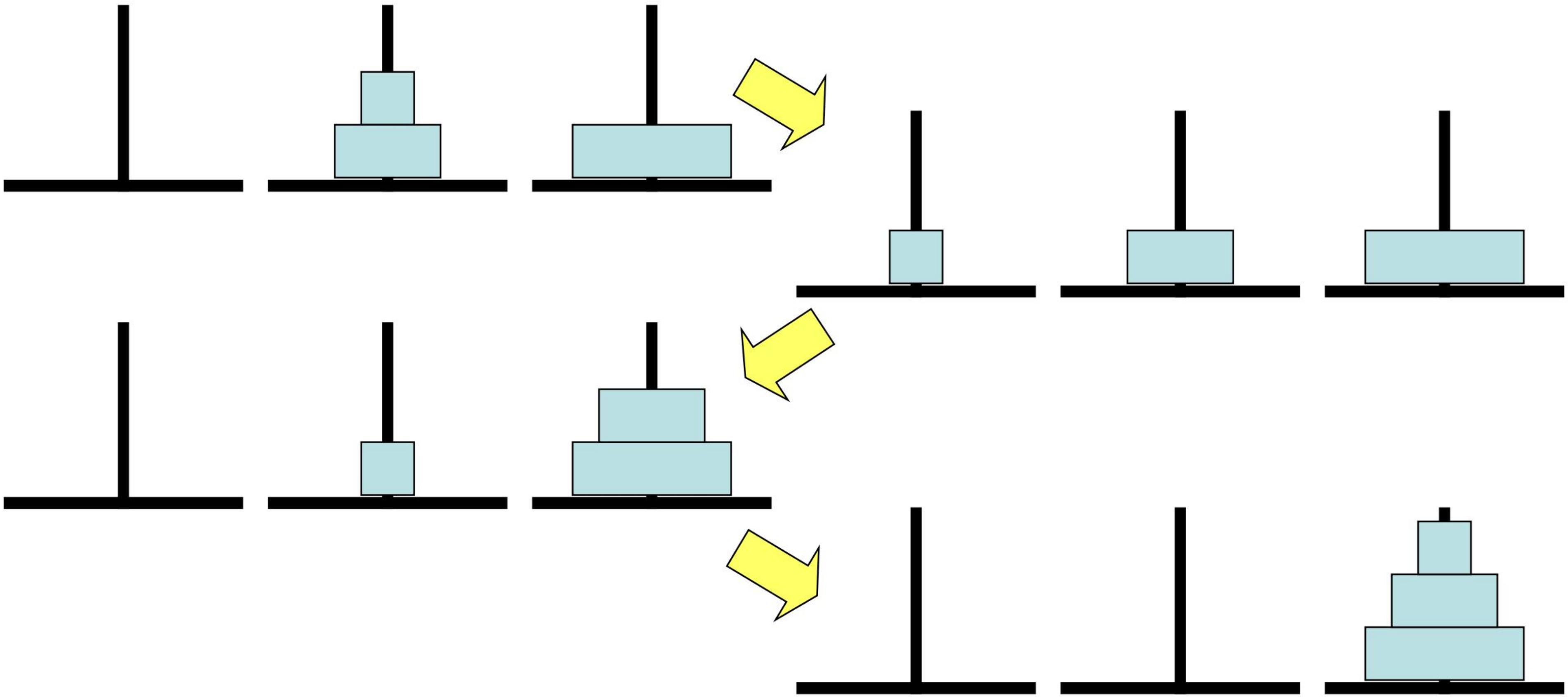
如果我们只有 3 个磁盘,这并不难。



在下一页继续

河内塔

如果我们只有 3 个磁盘,这并不难。



一共七步。

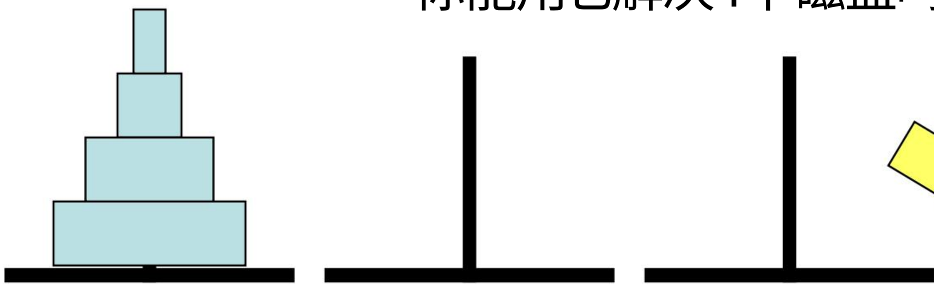
河内塔

你能写一个程序来解决这个问题吗?

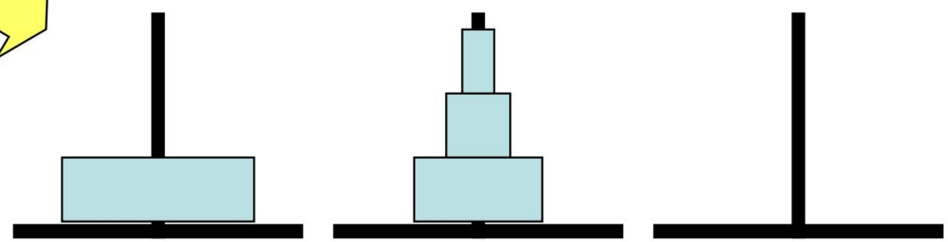
递归思考!

河内塔

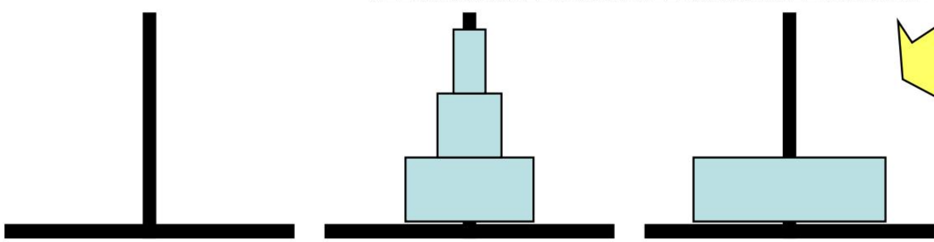
假设您已经有一个用于 3 个磁盘的程序。
你能用它解决4个磁盘吗?



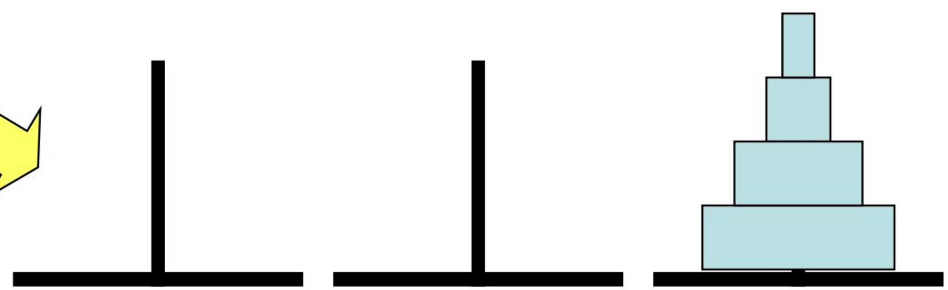
为了移动最大的磁盘,我必须先移动前 3 个磁盘,这样我就可以使用 3 个磁盘的程序。



然后我移动最大的磁盘。



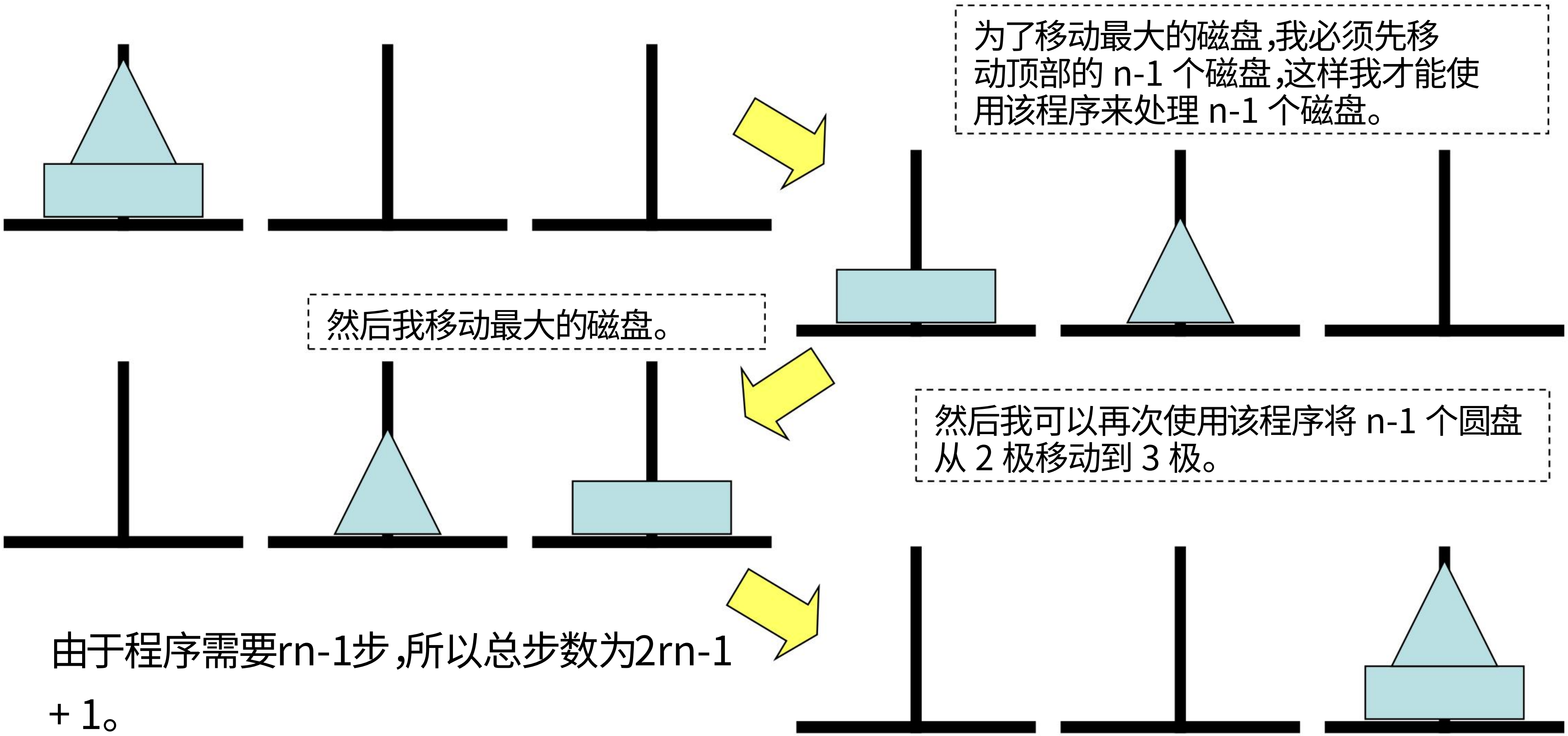
然后我可以使用程序将 3 个圆盘从 2 极移动到 3 极。



由于该程序需要 7 个步骤,因此总步骤数为 15。

河内塔

这种递归对于任何 n 都是正确的。



由于程序需要rn-1步,所以总步数为2rn-1 + 1。

rn = 2rn-1 + 1

最优性

如果我们有 4 个极点,最少需要多少步?

Frame-Stewart 算法是最优的。 (2014)

如果我们有 5 个极点,最少需要多少步?

如果我们有更多的极点怎么办?

打开问题…

这导致了“Frame-Stewart猜想”

注意 :原始拼图的另一种概括是将给定配置移动到另一个配置。这导致了一般的最短路径问题,这通常很难计算。

合并排序

给定一个由 n 个数字组成的序
列,需要多少步才能将它们排序为非递减顺序?

对数字进行排序的一种方法称为“冒泡排序”,其中每一步我们都会比较两个相邻的数字,如果它们是无序的则进行交换。

该算法可能需要大约n2/2 步。

例如,如果给定反向序列 n,n-1,n-2,⋯,1。

每次它都会搜索到最后找到最小的数,所以算法大致需要 (n-1)+(n-2)+⋯+1 = n(n-1)/2 步。

我们可以设计一个更快的算法吗?

递归思考!

合并排序

假设我们有一个对 n/2 个数字进行排序的程序。

我们可以使用它按以下方式对 n 个数字进行排序。

3 8 4 7 2 1 6 5

将序列分成两半。

使用该程序对两半进行独立排序。

3 4 7 8	1 2 5 6

通过这两个 n/2 数字的排序序列,我们可以轻松地将它们合并为 n 数字的排序序列!

合并排序

宣称。假设我们有两个k个数的排序序列。
我们可以用2k步将它们合并成 2k 个数字的
排序序列。

举例证明：

3 5 7 8 9 10

↑

1 2 4 6 11 12

↑

要确定两个序列中的最小数字，我们只需要查看两个序列的“头”即可。

因此，对于每一步，我们可以将排序后的序列扩展一个数字。

所以 2k 个数字的总步数是2k。

合并排序

3 5 7 8 9 10	1 2 4 6 11 12	1
3 5 7 8 9 10	1 2 4 6 11 12	1 2
3 5 7 8 9 10	1 2 4 6 11 12	1 2 3
3 5 7 8 9 10	1 2 4 6 11 12	1 2 3 4
3 5 7 8 9 10	1 2 4 6 11 12	1 2 3 4 5
3 5 7 8 9 10	1 2 4 6 11 12	1 2 3 4 5 6
3 5 7 8 9 10	1 2 4 6 11 12	1 2 3 4 5 6 7
3 5 7 8 9 10	1 2 4 6 11 12	1 2 3 4 5 6 7 8
3 5 7 8 9 10	1 2 4 6 11 12	1 2 3 4 5 6 7 8 9
3 5 7 8 9 10	1 2 4 6 11 12	1 2 3 4 5 6 7 8 9 10
3 5 7 8 9 10	1 2 4 6 11 12	1 2 3 4 5 6 7 8 9 10 11
3 5 7 8 9 10	1 2 4 6 11 12	1 2 3 4 5 6 7 8 9 10 11 12

合并排序

宣称。假设我们有两个k个数的排序序列。
我们可以用2k步将它们合并成 2k 个数字的排序
序列。

假设我们可以在Tk步中对 k 个数字进行排序。

然后我们可以用 2Tk + 2k 步对 2k 个数字进行排序。

因此， T2k = 2Tk + 2k。（如果 n 是奇数呢？）

如果我们解决这个重复（我们稍后会做），

然后我们将看到T2n ≤ n log2 n。

这比冒泡排序要快得多！

评论

这是“分而治之”算法的一个例子。

这个想法非常强大。

它可用于为一些基本问题设计更快的算法，例如整数乘法、矩阵乘法等。

计划

- 设置重复
  - 斐波那契重现
  - 解决问题的重复
  - 加泰罗尼亚语复发
- 解决重复问题

插入语

有多少种有效的方法来添加 n 对括号?

例如,添加 3 对括号有 5 种有效方法。

((())) (()())　　(())() ()(()) ()()()

设rn是添加 n 对括号的方法数。

我们如何使用r1 ,r2 ,...,rn-1 来计算它?

插入语

有多少种有效的方法来添加 n 对括号?

设rn是添加 n 对括号的方法数。

情况1: ()--------------------

rn-1种方法来添加剩余的 n-1 对。

所以在这种情况下有rn-1 。

案例2:(--) ------

1 种方式添加 1 对

r n-2种方法来添加剩余的 n-2 对。

所以在这种情况下有rn-2 。

案例3:(----)----------------

添加 2 对的 2 种方法

r n-3种方法来添加剩余的 n-3 对。

所以在这种情况下有2rn-3 。

插入语

有多少种有效的方法来添加 n 对括号？

设rn是添加 n 对括号的方法数。

案例k：$\underbrace{(\text{----------})}\ \underbrace{\text{----------}}$

r_{k-1}种方法来添加 k-1 对 r_{nk}种方法来添加剩余的 nk 对。

根据乘积法则,情况 k 有rk-1 rn-k种方式。

这些案例按第一个左括号的右括号的位置排序,因此这些案例是不相交的。

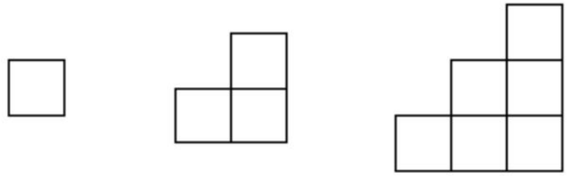
因此,根据求和规则,

$$r_n = \sum_{k=1}^{n} r_{k-1} r_{n-k}$$

楼梯

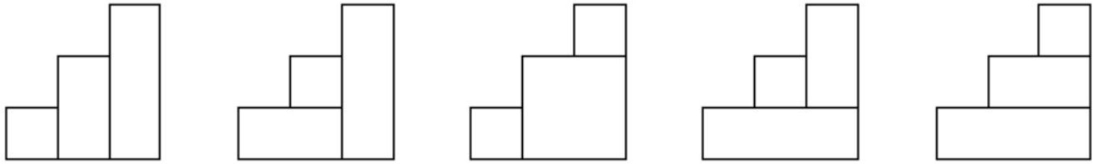
n 阶梯是由 x 轴、y=x 和 x=n+1 界定的单位正方形的集合。

例如 1-stair、2-stair 和 3-stair 是这样的：



有多少种方法可以用 n 个矩形填充 n 级楼梯?

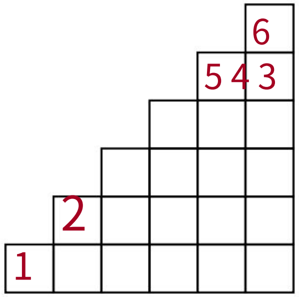
例如,有 5 种方法可以用 3 个矩形填充 3 个楼梯。



楼梯

设rn是用 n 个矩形填充 n 级楼梯的方法数。

我们如何使用r1 ,r2 ,...,rn-1 来计算它?



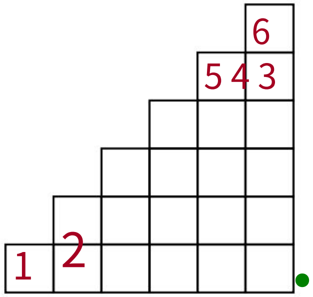
给定 n 楼梯,第一个观察结果是对角线上的位置 (红色数字)必须被不同的矩形覆盖。

由于对角线上有 n 个位置,并且我们只能使用 n 个矩形,因此每个矩形必须恰好覆盖一个红色数字。

楼梯

设rn是用 n 个矩形填充 n 级楼梯的方法数。

我们如何使用r1 ,r2 ,...,rn-1 来计算它?



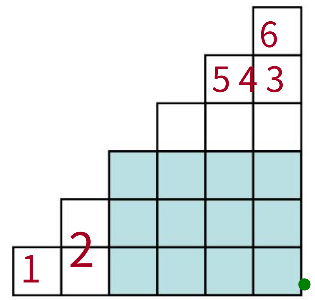
考虑覆盖右下角的矩形R （用o 标记）。

我们根据R覆盖的红色数字考虑不同的情况。

楼梯

设rn是用 n 个矩形填充 n 级楼梯的方法数。

我们如何使用r1 ,r2 ,...,rn-1 来计算它?



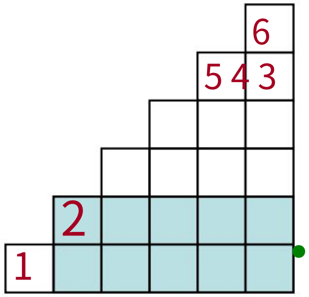
假设R覆盖3。那么 6 级楼梯被分成 3 个部分。
一部分是矩形R。另外两部分是 2 级楼梯和 3 级楼梯。

因此,在这种情况下,剩余部分的填充方式数量等于r2 r3

楼梯

设rn是用 n 个矩形填充 n 级楼梯的方法数。

我们如何使用r1 ,r2 ,...,rn-1 来计算它?



同样假设R覆盖2。

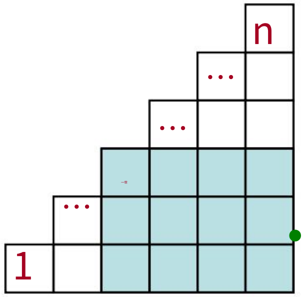
然后矩形将楼梯“打破”为 1 级楼梯和 4 级楼梯。

因此,在这种情况下,剩余部分的填充方式数量等于r1 r4

楼梯

设rn是用 n 个矩形填充 n 级楼梯的方法数。

我们如何使用r1 ,r2 ,...,rn-1 来计算它?

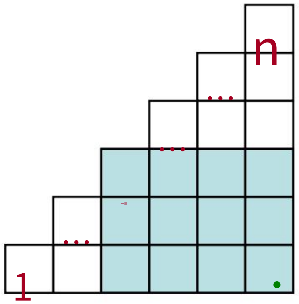


一般来说,假设矩形覆盖i然后矩形将楼梯“打破”为 (i-1)-stair 和 (ni)-stair。

因此,在这种情况下,剩余部分的填充方式的数量等于ri-1 rn-i （我们定义r0=1）

楼梯

剩余部分的填充方式数等于ri-1 rn-i



矩形R覆盖不同的i会形成不同的配置,每个配置必须对应其中一种情况。

因此总路数等于

$$r_n = \sum_{i=1}^{n} r_{i-1} r_{n-i}$$

加泰罗尼亚语号码

有多少种有效的方法来添加 n 对括号？

$$r_n = \sum_{k=1}^{n} r_{k-1} r_{n-k}$$

所以楼梯问题的递归与括号问题的递归相同。可以证明

$$r_n = \frac{1}{n+1}\binom{2n}{n}$$

这被称为第n 个加泰罗尼亚数。

计划

- 设置重复
  - 斐波那契重现
  - 解决问题的重复
  - 加泰罗尼亚语复发
- 解决重复问题

暖身

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

$a_1 = a_0 + 2$

$a_2 = a_1 + 2 = (a_0 + 2) + 2 = a_0 + 4$

$a_3 = a_2 + 2 = (a_0 + 4) + 2 = a_0 + 6$

$a_4 = a_3 + 2 = (a_0 + 6) + 2 = a_0 + 8$

看到模式是$a_k = a_0 + 2k = 2k+1$

你可以通过归纳来验证。

求解河内序列

a1=1,   ak = 2ak-1 + 1

a2 = 2a1 + 1 = 3

a3 = 2a2 + 1 = 2(2a1 + 1) + 1 = 4a1 + 3 = 7

a4 = 2a3 + 1 = 2(4a1 + 3) + 1 = 8a1 + 7 = 15

a5 = 2a4 + 1 = 2(8a1 + 7) + 1 = 16a1 + 15 = 31

a6 = 2a5 + 1 = 2(16a1 + 15) + 1 = 32a1 + 31 = 63

猜猜模式是ak = 2k-1

你可以通过归纳来验证。

解决合并排序递归

$$T2k = 2Tk + 2k$$

如果我们可以猜到Tk = k log2k,那么我们可以证明T2k = 2k log2 (2k)。

这是因为T2k = 2Tk + 2k

$$= 2k\log2k + 2k$$

$$= 2k(\log2k + 1)$$

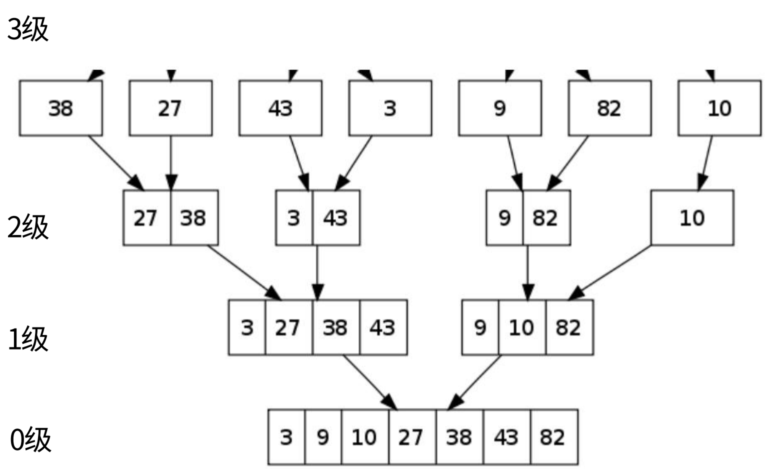
$$= 2k(\log2k + \log22)$$

$$= 2k\log22k$$

解决合并排序递归

$T2k = 2Tk + 2k$

我们怎么能猜到Tk = k log2k?



Tk ≤ log2k   k = klog2k

#级别

#comparisons 需要每 1 个

= #steps 需要在每个合并问题中对数字进行排序

当 k 是 2 的幂时,实现相等。

求解斐波那契数列

$a0=0, a1=1,$ $an = an-1 + an-2$

$a2 = a1 + a0 = 1$

$a3 = a2 + a1 = 2$

$a4 = a3 + a2 = 3$

$a5 = a4 + a3 = 5$

$a6 = a5 + a4 = 8$

$a7 = a6 + a5 = 13$

...

我们如何找到一个公式?

生成函数

$$\langle 0,0,0,0,\ldots\rangle \leftrightarrow 0+0x+0x^2+0x^3+\ldots=0$$

$$\langle 1,1,1,1,\ldots\rangle \leftrightarrow 1+x+x^2+x^3+\ldots \qquad =1/(1-x)$$

$$\langle 1,-1,1,-1,\ldots\rangle \leftrightarrow 1-x+x^2-x^3+\ldots \qquad =1/(1+x)$$

$$\boxed{\langle a_0,a_1,a_2,a_3,\ldots\rangle \leftrightarrow F(x)}$$

这称为{an }的普通生成函数：

F(x)=a0 +a1x+a2x 2+a3x 3+⋯

sequence ↔ generating function

生成函数

缩放：

$$\langle f_0, f_1, f_2, \ldots \rangle \longleftrightarrow F(x) \quad \Rightarrow \quad \langle cf_0,\ cf_1,\ cf_2,\ \ldots \rangle \longleftrightarrow c \cdot F(x)$$

添加：

$$\begin{array}{rrcl} & \langle f_0, f_1, f_2, \ldots \rangle & \longleftrightarrow & F(x) \\ + & \langle g_0, g_1, g_2, \ldots \rangle & \longleftrightarrow & G(x) \\ \hline \multicolumn{2}{r}{\langle f_0 + g_0,\ f_1 + g_1,\ f_2 + g_2,\ \ldots \rangle} & \longleftrightarrow & F(x) + G(x) \end{array}$$

右移：

$$\langle \underbrace{0, 0, \ldots, 0}_{k \text{ zeroes}}, f_0, f_1, f_2, \ldots \rangle \longleftrightarrow x^k \cdot F(x)$$

差异化：

$$\langle f_0, f_1, f_2, f_3, \ldots \rangle \longleftrightarrow F(x) \quad \Rightarrow \quad \langle f_1, 2f_2, 3f_3, \ldots \rangle \longleftrightarrow F'(x)$$

生成函数

我们如何找到 <0,1,4,9,...> 的生成函数?

$$\langle 1,1,1,1,\dots\rangle \longleftrightarrow \frac{1}{1-x}$$

$$\langle 1,2,3,4,\dots\rangle \longleftrightarrow \frac{d}{dx}\frac{1}{1-x} = \frac{1}{(1-x)^2}$$

$$\langle 0,1,2,3,\dots\rangle \longleftrightarrow x\cdot\frac{1}{(1-x)^2} = \frac{x}{(1-x)^2}$$

$$\langle 1,4,9,16,\dots\rangle \longleftrightarrow \frac{d}{dx}\frac{x}{(1-x)^2} = \frac{1+x}{(1-x)^3}$$

$$\langle 0,1,4,9,\dots\rangle \longleftrightarrow x\cdot\frac{1+x}{(1-x)^3} = \frac{x(1+x)}{(1-x)^3}$$

所以生成函数是

$$\boxed{\frac{x(1+x)}{(1-x)^3}}$$

求解斐波那契数列

生成函数如何帮助解决斐波那契数列?

斐波那契数列： f0=0, f1=1, fn = fn-1 + fn-2

所以 {fn }的生成函数是

F(x)=f0 +f1x+f2x 2+f3x 3+⋯=0+x+(f1 +f0)x2+(f2 +f1)x3+⋯

我们可以找到 <0,1,f1 +f0 ,f2 +f1 ,...> 的生成函数!

$$
\begin{array}{cccccccccc}
 & \langle & 0, & 1, & 0, & 0, & 0, & \ldots & \rangle & \longleftrightarrow & x \\
 & \langle & 0, & f_0, & f_1, & f_2, & f_3, & \ldots & \rangle & \longleftrightarrow & xF(x) \\
+ & \langle & 0, & 0, & f_0, & f_1, & f_2, & \ldots & \rangle & \longleftrightarrow & x^2F(x) \\
\hline
 & \langle & 0, & 1+f_0, & f_1+f_0, & f_2+f_1, & f_3+f_2, & \ldots & \rangle & \longleftrightarrow & x+xF(x)+x^2F(x)
\end{array}
$$

F(x)=x+xF(x)+x2F(x)

求解斐波那契数列

解决 F(x) 我们得到：

F(x)=x/(1-xx 2)

部分分馏给出：

$$F(x) = \frac{1}{\sqrt{5}}\left(\frac{1}{1-\alpha_1 x} - \frac{1}{1-\alpha_2 x}\right)$$

where $\alpha_1 = \frac{1}{2}(1+\sqrt{5})$ and $\alpha_2 = \frac{1}{2}(1-\sqrt{5})$.

使用泰勒级数展开 F(x)：

$$F(x) = \frac{1}{\sqrt{5}}\left((1+\alpha_1 x+\alpha_1^2 x^2+\cdots) - (1+\alpha_2 x+\alpha_2^2 x^2+\cdots)\right)$$

$$\boxed{f_n = \frac{\alpha_1^n - \alpha_2^n}{\sqrt{5}} = \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right)}$$

二阶递归关系

似乎我们做了一些更一般的事情,比如:

$a_k = Aa_{k-1} + Ba_{k-2}$

A和B是实数,B≠0

这被称为“具有常数系数的二阶线性齐次递推关系”。

例如,斐波那契数列是当 A=B=1 时。

我们可以对这个问题给出一个一般性的答案吗?

异根定理

假设一个序列 $(a_0, a_1, a_2, a_3, \cdots)$ 满足递归关系$a_k = Aa_{k-1} + Ba_{k-2}$

如果$t^2 - At - B = 0$ 有两个不同的根 r 和 s，

那么对于某些 C 和 D， $a_n = Cr^n + Ds^n$ 。

该定理说,递归关系的任何解都是两个级数$(1,r,r^2,r^3,r^4,\ldots,r^n,\ldots)$ 和 $(1,s,s^2,s^3,s^4,\ldots,s^n,\cdots)$ 由$t^2 - At - B = 0$ 的不同根定义。

因此,如果给定a_0和a_1 ,则 C 和 D 是唯一确定的。

例子

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

需要找到 (1, t, t2, t3, t4, ..., , ...) t n形式的解

其中 t 是二次方程t2 - t – 2 = 0 的根。

这意味着 r=2 或 s=-1。

如果我们不知道a0， a1 ,它们是 {an } 的许多解决方案：

(i) (1,2,4,8,16,32,64,···) 一个=rn

(ii) (1,-1,1,-1,1,-1,···) an=sn

(iii) (2,1,5,7,17,31,65,···) (iv)··· an =r n+sn

···

重温斐波那契数列

如果给定a0 ,a1 ...

a0=0, a1=1, an = an-1 + an-2

首先求解二次方程t2 - t – 1 = 0。

$$t = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

所以不同的根源是：

$$r = \frac{1+\sqrt{5}}{2} \qquad s = \frac{1-\sqrt{5}}{2}$$

重温斐波那契数列

a0=0, a1=1, an = an-1 + an-2

根据异根定理,解满足以下公式:

$$a_n = C(\frac{1+\sqrt{5}}{2})^n + D(\frac{1-\sqrt{5}}{2})^n$$

为了计算出 C 和 D,我们将a0和a1的值代入:

$$0 = C + D$$

$$1 = C(\frac{1+\sqrt{5}}{2}) + D(\frac{1-\sqrt{5}}{2})$$

重温斐波那契数列

解这两个方程,我们得到:

$$C = \frac{1}{\sqrt{5}}, D = -\frac{1}{\sqrt{5}}$$

所以:

$$a_n = C(\frac{1+\sqrt{5}}{2})^n + D(\frac{1-\sqrt{5}}{2})^n$$

$$= \frac{1}{\sqrt{5}}(\frac{1+\sqrt{5}}{2})^n - \frac{1}{\sqrt{5}}(\frac{1-\sqrt{5}}{2})^n$$

快速总结

递归是计算机科学中非常有用的技术。

通过将问题简化为更小的问题,学习递归思考非常重要。

如果一个人想成为一名专业的程序员,这是一项必不可少的技能。

确保您在设置递归关系和生成函数方面有更多的练习。