22.2 递推方程的其他解法

- 换元法
- 迭代归纳法--递归树
- 差消法
- 尝试法
- 应用实例

换元法

思想：通过换元转化成常系数线性递推方程

例 1 $\begin{cases} a_n^2 = 2a_{n-1}^2 + 1 \\ a_0 = 2 \end{cases}$ $a_n > 0$

令 $b_n = a_n^2$，代入得

$b_n = 2\,b_{n\text{-}1} + 1$，$b_0 = 4$

解得 $b_n = 5 \cdot 2^n - 1$，$a_n = \sqrt{5 \cdot 2^n - 1}$

换元法—归并排序

例 2　归并排序

$T(n) = 2\,T(\,n/2\,) + n\text{-}1\,,\quad n = 2^k$

$T(2) = 1$

解　$H(k) = 2\,H(k\text{-}1) + 2^k - 1$

$H(1) = 1$

令 $H^*(k) = P_1 k 2^k + P_2$，解得 $P_1 = P_2 = 1$,

$H^*(k) = k2^k + 1$

通解 $H(k) = C\,2^k + k2^k + 1$,

代入初值，得 $C = -1$,

$H(k) = -\,2^k + k2^k + 1$,

$T(n) = n \log n - n + 1$

迭代归纳法

例 3 计数 $a_1, a_2, \ldots, a_n$ 相乘（可交换）的方法数

$$h(n) = (4n\text{-}6)\, h(n\text{-}1)$$

$$h(1) = 1$$

$$\begin{aligned} h(n) &= (4n-6)\, h(n-1) \\ &= (4n-6)(4n-10)\, h(n-2) \\ &= \ldots \\ &= (4n-6)(4n-10) \ldots 6 \cdot 2 \cdot h(1) \\ &= 2^{n-1}[(2n-3)(2n-5) \ldots 3 \cdot 1] \\ &= 2^{n-1} \frac{(2n-2)!}{(2n-2)(2n-4) \ldots 4 \cdot 2} = \frac{(2n-2)!}{(n-1)!} \end{aligned}$$

用归纳法验证.

迭代归纳法—错位排列

例 4　错位排列问题

错位排列：$\{1,2,\ldots,n\}$的排列 $a_1a_2\ldots a_n$, $a_i\neq i$, $i=1,2,\ldots,n$，

n 个元素的错位排列数记作 D_n

将错位排列按首元素 $2,3,\ldots,n$ 分类：有 n-1 类，

第一位为 2 的类：

第二位为 1：　方法数为 D_{n-2}

第二位不是 1：方法数为 D_{n-1}

递推方程：

$D_n=(n-1)(D_{n-1}+D_{n-2})$

$D_1=0$, $D_2=1$

迭代归纳法—错位排列

解：$D_n=(n\text{-}1)(D_{n\text{-}1}+D_{n\text{-}2})$

$$D_n-nD_{n-1}=-[D_{n-1}-(n-1)D_{n-2}]=\ldots$$

$$=(-1)^{n-2}[D_2-2D_1]=(-1)^{n-2}$$

$$D_n=nD_{n-1}+(-1)^n,\quad D_1=0$$

$$D_n=n(n-1)D_{n-2}+n(-1)^{n-1}+(-1)^n$$

$$=n(n-1)(n-2)D_{n-3}+n(n-1)(-1)^{n-2}+n(-1)^{n-1}+(-1)^n$$

$$=\ldots$$

$$=n(n-1)\ldots 2D_1+n(n-1)\ldots 3(-1)^2$$

$$+n(n-1)\ldots 4(-1)^3+\ldots+n(-1)^{n-1}+(-1)^n$$

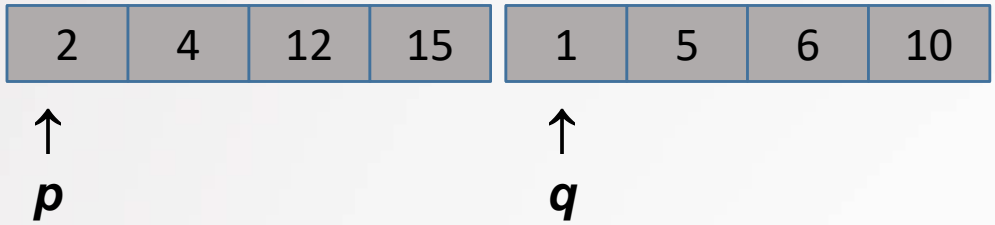
$$=n![1-\frac{1}{1!}+\frac{1}{2!}-\ldots+(-1)^n\frac{1}{n!}]$$

迭代归纳法—递归树

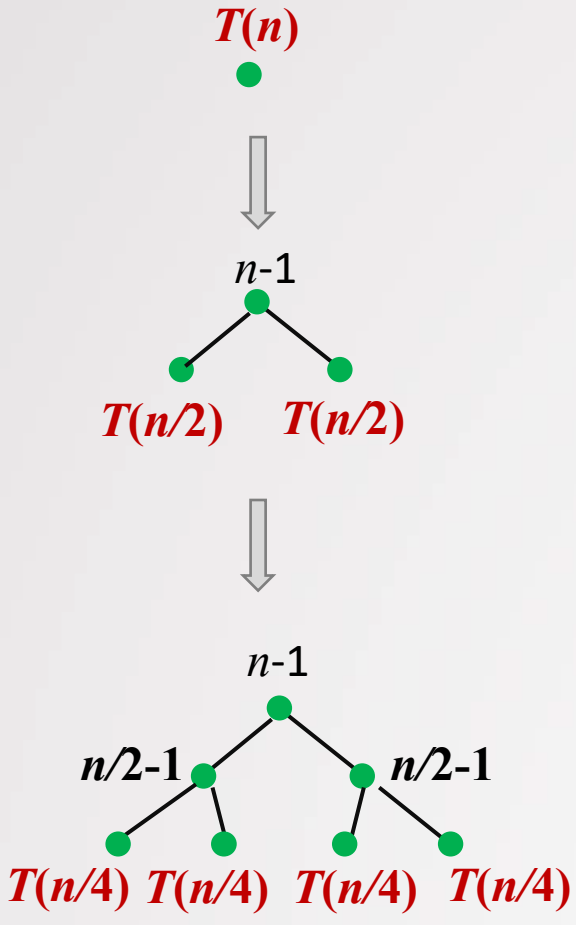
例 5　归并排序

$$T(n) = 2T(n/2) + n\text{-}1, \qquad n=2^k$$

$$T(2) = 1$$

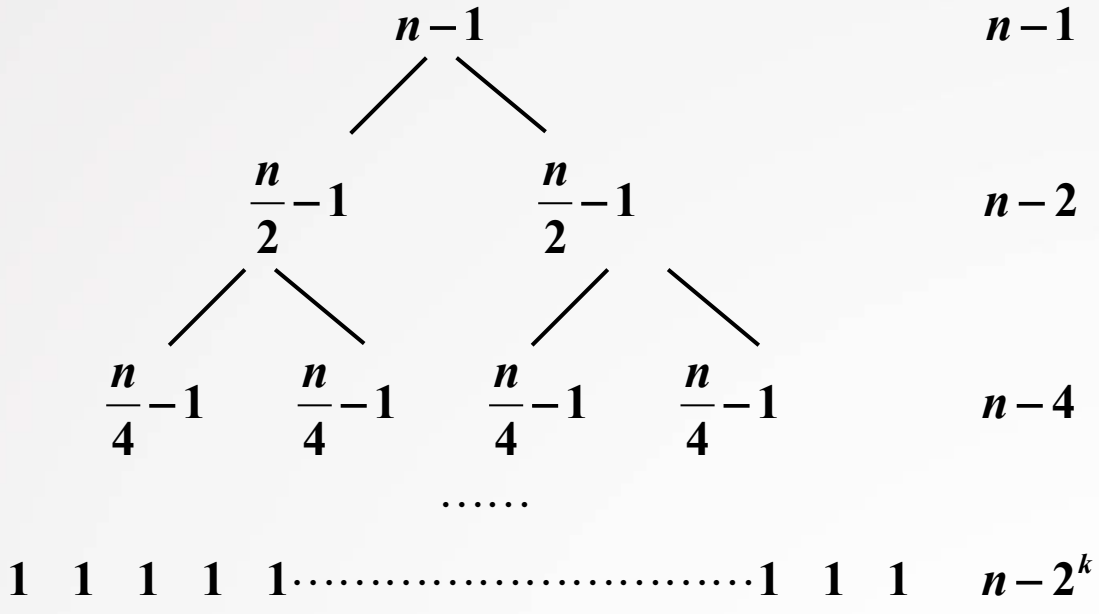


迭代模型：递归树



生成过程

$$T(n)=2T(n/2)+n-1,\ n=2^k,\quad T(1)=0$$



$$T(n)=n-1+n-2+\ldots+n-2^k$$

$$=kn-(2^k-1)=n\log n-n+1$$

差消法—快速排序

例 6　快速排序

算法　快速排序 Quicksort

输入：数组 $A[p..r]$

输出：排好序的数组 A

Quicksort(A,p,r)

1. if $p<r$
2. then $q \leftarrow$Partition(A,p,r)
3. $A[p] \leftrightarrow A[q]$
4. Quicksort(A,p,q-1)
5. Quicksort(A,q+1,r)

平均情况下复杂度分析

$$\begin{cases} T(n)=\dfrac{2}{n}\sum_{i=1}^{n-1}T(i)+n+1, & n\geq 2 \\ T(1)=0 \end{cases}$$

$$nT(n)=2\sum_{i=1}^{n-1}T(i)+n^2+n$$

$$(n-1)T(n-1)=2\sum_{i=1}^{n-2}T(i)+(n-1)^2+(n-1)$$

$$nT(n)-(n-1)T(n-1)=2T(n-1)+2n$$

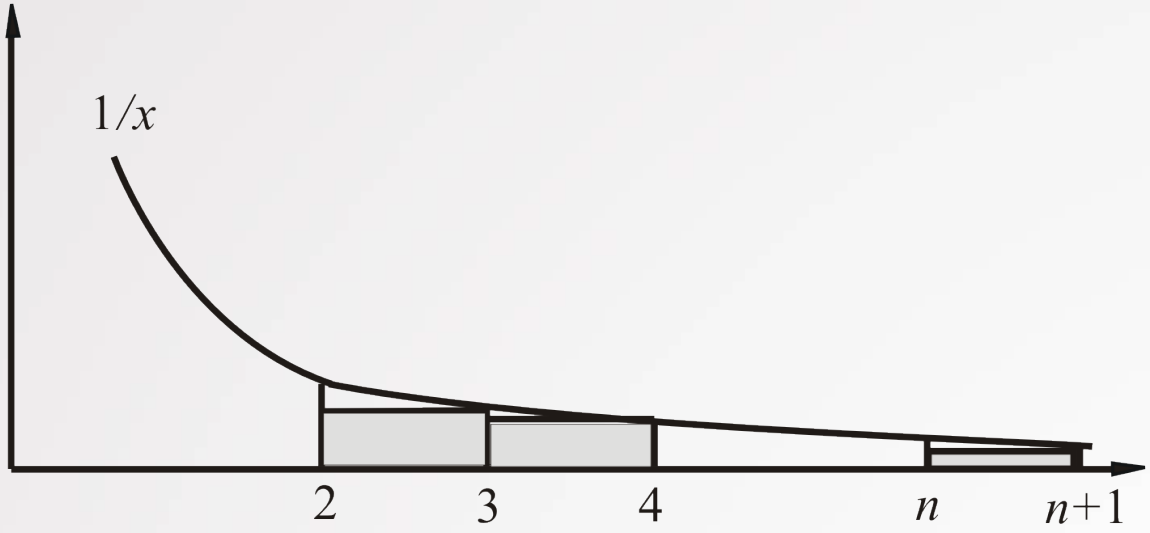
$$nT(n)=(n+1)T(n-1)+2n$$

$$\frac{T(n)}{n+1}=\frac{T(n-1)}{n}+\frac{2}{n+1}=\frac{2}{n+1}+\frac{2}{n}+\ldots+\frac{2}{3}+\frac{T(1)}{2}$$

$$=2\left[\frac{1}{n+1}+\frac{1}{n}+\ldots+\frac{1}{3}\right]=O(\log n)$$

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

积分近似



$$\frac{1}{n+1}+\frac{1}{n}+\ldots+\frac{1}{3}\le\int_{2}^{n+1}\frac{1}{x}dx$$

$$=\ln x\Big|_{2}^{n+1}=\ln(n+1)-\ln 2=O(\log n)$$

分治算法

n 为输入规模，n/b 为子问题输入规模，

a 为子问题个数，$d(n)$为分解及综合的代价

$$T(n)=aT(n/b)+d(n),\quad n=b^k$$

$$T(1)=1$$

$$T(n)=a^2T(n/b^2)+ad(n/b)+d(n)=\ldots$$

$$=a^kT(n/b^k)+a^{k-1}d(n/b^{k-1})+a^{k-2}(n/b^{k-2})+$$

$$\ldots+ad(n/b)+d(n)$$

$$=a^k+\sum_{i=0}^{k-1}a^id(n/b^i)$$

$$a^k=a^{\log_b n}=n^{\log_b a}$$

分治与递归算法—二分检索

$$T(n)=a^k+\sum_{i=0}^{k-1}a^i d(n/b^i),\quad a^k=n^{\log_b a}$$

(1) $d(n)=c$

$$T(n)=\begin{cases}a^k+c\dfrac{a^k-1}{a-1}=O(a^k)=O(n^{\log_b a}) & a\neq 1\\ a^k+kc=O(kc)=O(\log n) & a=1\end{cases}$$

二分检索

$$W(n)=W(n/2)+1$$

$$a=1,\ b=2,\ d(n)=c$$

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

分治与递归算法—归并排序

(2) $d(n)=cn$

$$T(n)=a^k+\sum_{i=1}^{k-1}a^i\frac{cn}{b^i}=a^k+cn\sum_{i=1}^{k-1}(\frac{a}{b})^i$$

$$=\begin{cases} n^{\log_b a}+cn\dfrac{(a/b)^k-1}{a/b-1}=O(n) & a<b \\ n+cnk=O(n\log n) & a=b \\ a^k+cn\dfrac{(a/b)^k-1}{a/b-1}=a^k+c\dfrac{a^k-b^k}{a/b-1}=O(n^{\log_b a}) & a>b \end{cases}$$

归并排序

$W(n)=2W(n/2)+n-1$

$a=2, b=2, d(n)=O(n),$

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