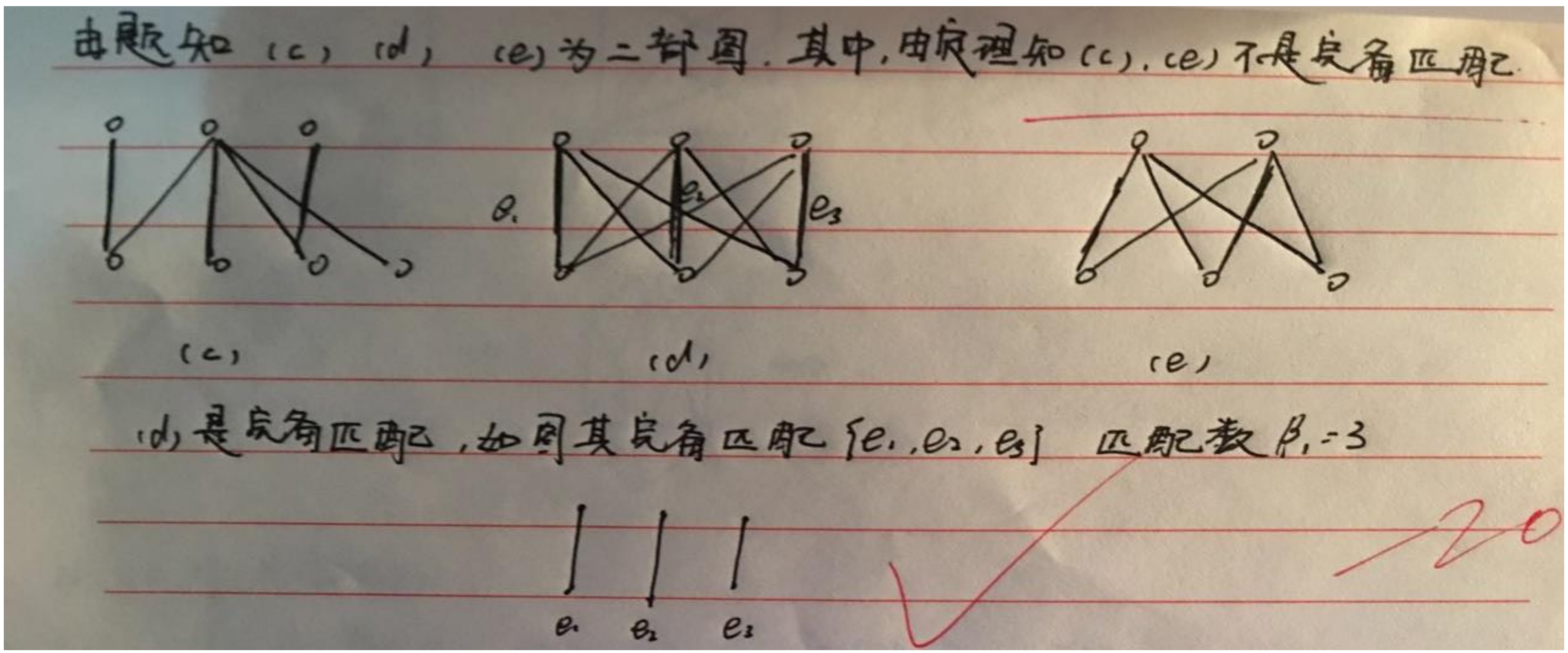
评分标准

- 第一题30分，三小题，每小题10分。画出二部图并画出完备匹配8分，匹配数2分
- 第二题20分：构造出有向图10分，写出一条欧拉回路5分，结果序列5分
- 第三题30分：三小题，每小题10分。通过库拉图斯基定理证明，需要给出收缩和消去的过程，否则扣5分。通过欧拉公式及推论等证明，需要写出完整的证明过程，否则0分。
- 第四题20分
- 附加题20分：定理6.7是哈密顿图的充分不必要条件，不能用来证明该图不是哈密顿图。该方法不给分

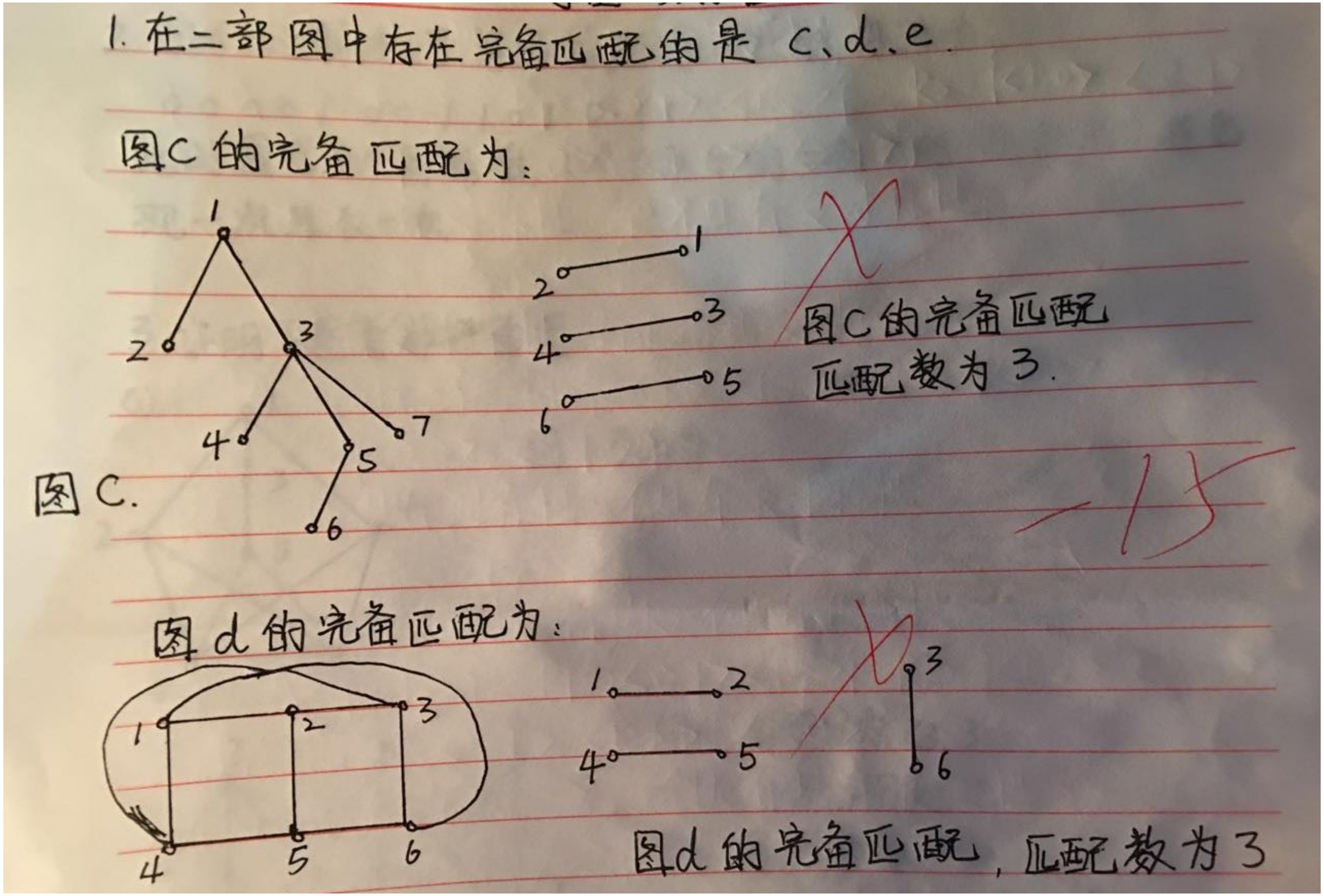
这次作业整体情况一般，有抄袭现象

- 第一题

a) 有同学错误用了定理**6.3**，推出了**c**，**e**不是完备匹配。**6.3**是完备匹配的充分不必要条件

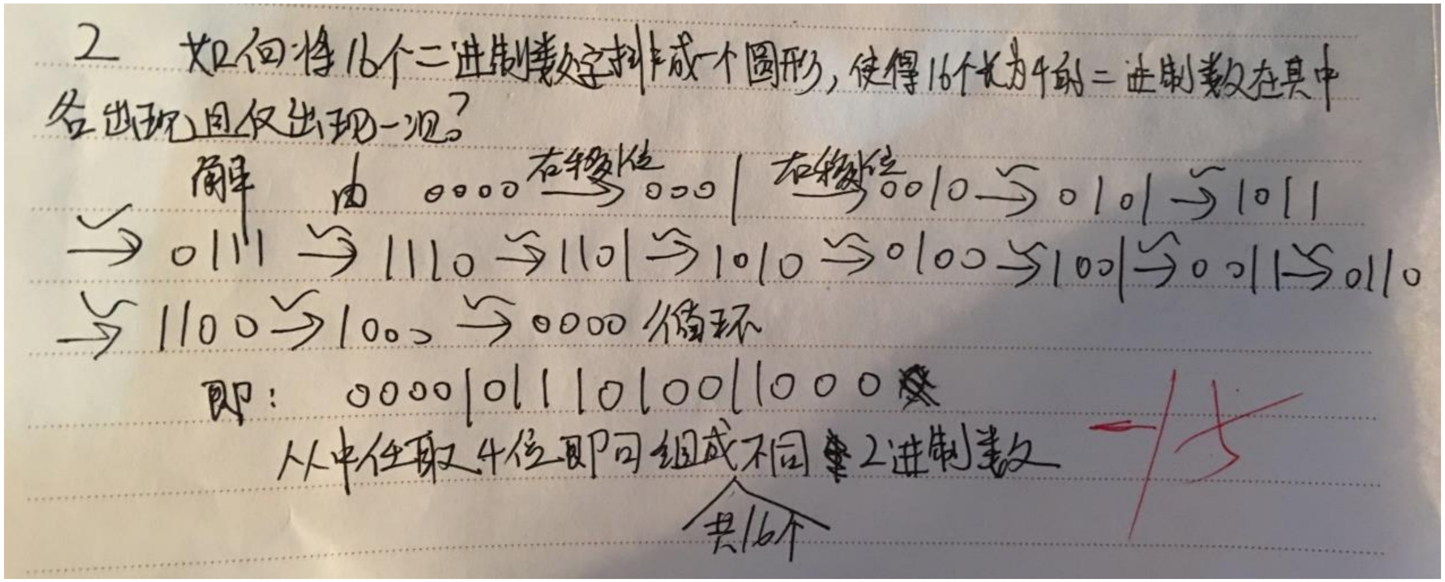


b) 完备匹配画法错误


1. 在二部图中存在完备匹配的是 c、d、e.
图C的完备匹配为：
图C.
图C的完备匹配
匹配数为3.
-15
图d的完备匹配为：
图d的完备匹配，匹配数为3

- 第二题，

a) 对题目理解有误，方法错误

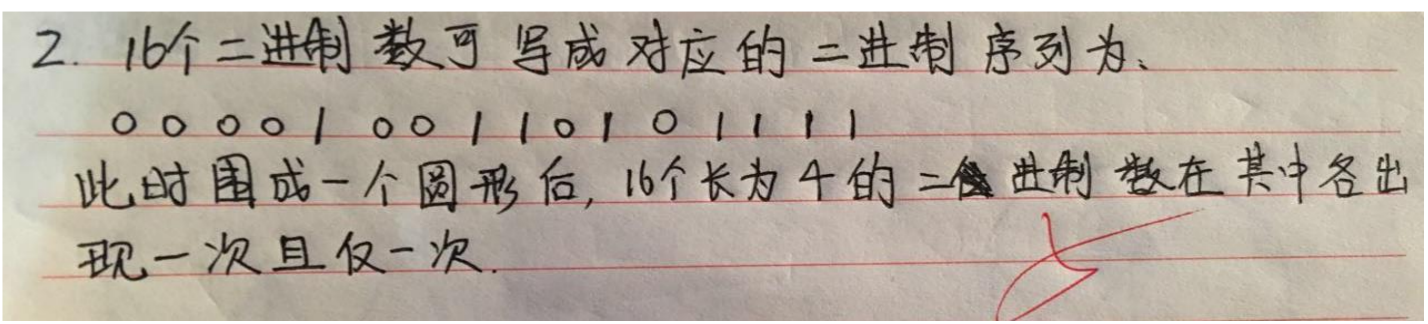

2 如何将16个二进制数字排成一个圆形，使得16个长为4的二进制数在其中各出现且仅出现一次？

解 由 0000 $\xrightarrow{右移1位}$ 0001 $\xrightarrow{右移1位}$ 0010 → 0101 → 1011 → 0111 → 1110 → 1101 → 1010 → 0100 → 1001 → 0011 → 0110 → 1100 → 1000 → 0000 循环

即：0000101110100110000

从中任取4位即可组成不同2进制数 共16个

b) 未画出有向图，未给出一条欧拉回路，亦或者没有任何过程。

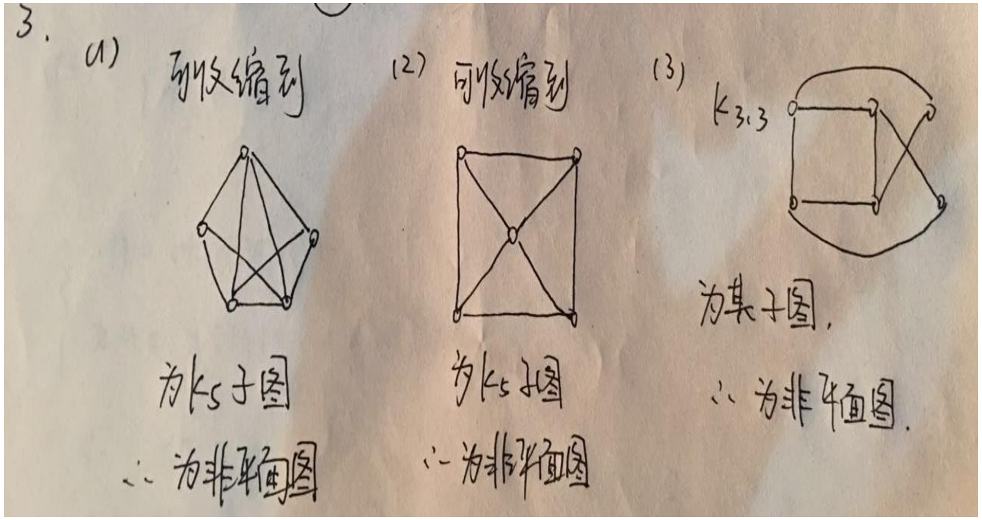

2. 16个二进制数可写成对应的二进制序列为、

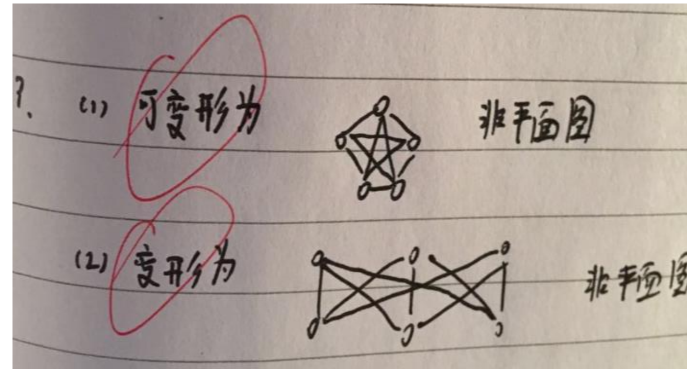
0000100110101111

此时围成一个圆形后，16个长为4的二进制数在其中各出现一次且仅一次.

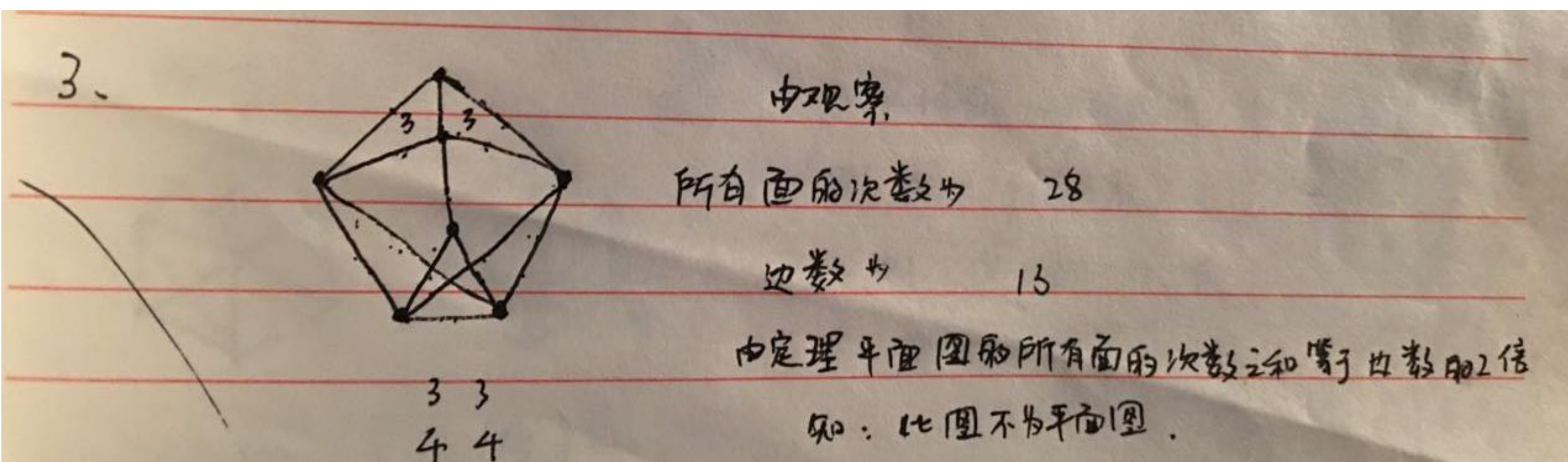
- 第三题

a) 没有给出收缩和消去的过程，并且证明方法错误，对库拉图斯基定理的理解错误





b）错误使用定理6.9



- 附加题

定理6.7是哈密顿图的充分不必要条件，并不能用来证明该图不是哈密顿图

附加题：设G是 $n(n\geq3)$ 阶无向简单图，若任意两个不相邻顶点度数和 $\geq n-1$，则G中存在哈密顿通路。

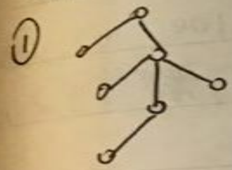
图中 $n=13$，任何一个不相邻顶点度数之和最大为 $8\leq13$

所以图为非哈密顿图。

范例

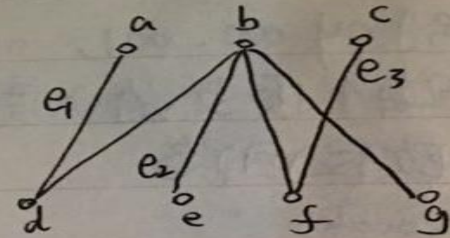
P151 例6.8

6个图中只有(c)(d)(e)是二部图。

① 

(c)

可画成

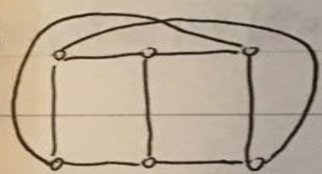


$V_1=\{a,b,c\}$

$V_2=\{d,e,f,g\}$

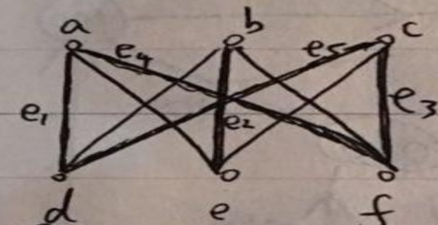
$M=\{e_1,e_2,e_3\}$是最大匹配，匹配数为3

又$|V_1|=3$，所以M是(c)中V_1到V_2的完备匹配。

② 

(d)

可画成



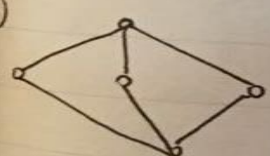
$V_1=\{a,b,c\}$

$V_2=\{d,e,f\}$

$M_2=\{e_2,e_4,e_5\}$

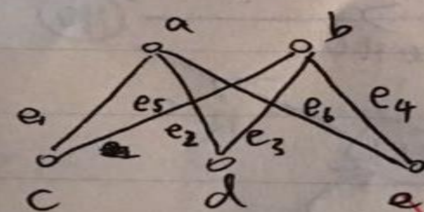
$M_1=\{e_1,e_2,e_3\}$是最大匹配，匹配数为3

又$|V_1|=3$，所以M_1和M_2是(d)中V_1到V_2的完备匹配。

③ 

(e)

可画成



$V_1=\{a,b\}$

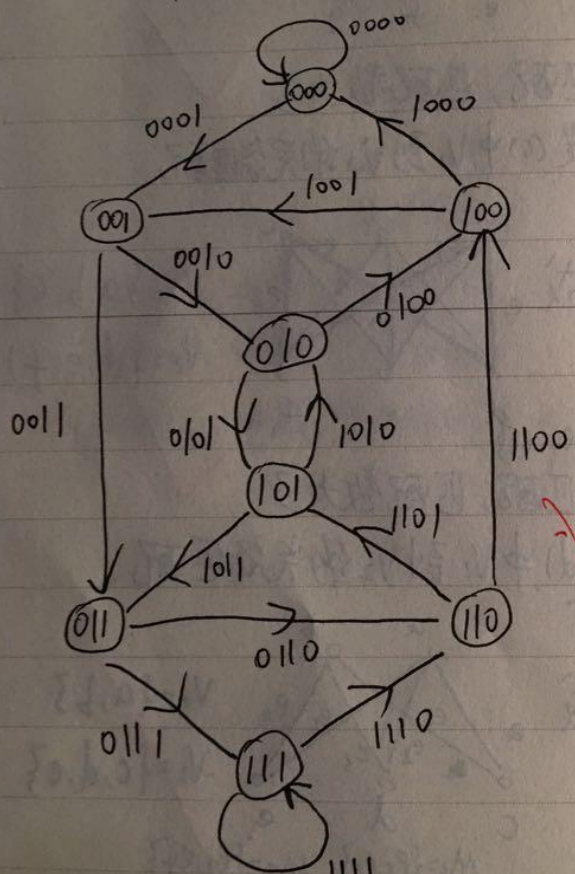
$V_2=\{c,d,e\}$

$M_3=\{e_5,e_6\}$，$M_4=\{e_1,e_4\}$

$M_1=\{e_1,e_3\}$，$M_2=\{e_2,e_4\}$是最大匹配，匹配数为2

又$|V_1|=2$，所以M_1，M_2和M_3，M_4是(e)中V_1到V_2的完备匹配。

2. 构造一个由16个0和1组成的圆环，使得圆环上连续4个数字的序列都不相同。为此，构造一个8阶有向图，8个顶点分别叫000，001，010，011，100，101，110，111。图是连通的且每个顶点的入度等于出度，都等于2，所以图中存在一条欧拉回路

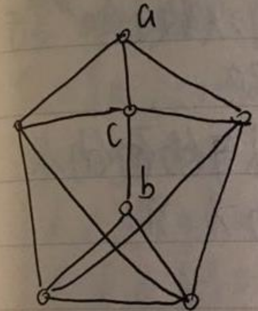


从abcd到bcd的边记为abcd，
如100到001记为1001.

其中一条欧拉回路为0000，0001，0011，0111，1111，1110，1100，1001，0010，0101，1011，0110，1101，1010，0100，1000.

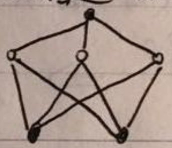
所以圆环序列为0000 1111 0010 1101，下一位为序列首位的0。

3. 证明下面各图为非平面图.

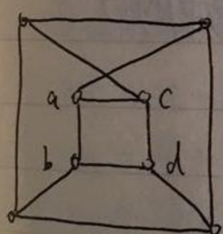


~~消去2度顶点c，再收缩边a，b，得到K_5~~

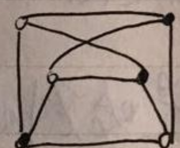
收缩边b，c，可取子图



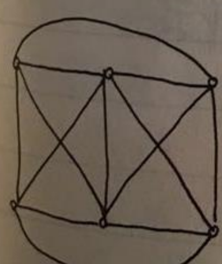
正是$K_{3,3}$.



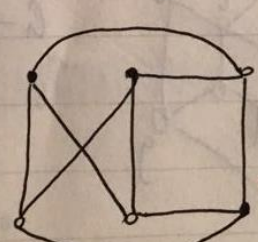
收缩边ab，收缩边cd，有子图



正是$K_{3,3}$



其子图



正是$K_{3,3}$

据库拉图斯基定理，以上三个图为非平面图

4. 7阶15条边的图G为简单连通平面图，证明其为极大平面图。

证：据欧拉公式，$n=7$，$m=15$，$7-15+r=2$

面数 $r=10$.

据定理6.9 平面图的所有面的次数之和等于边数2倍，所以所有面次数为30.

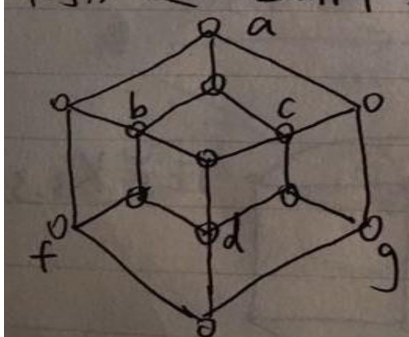
因为G是简单连通平面图，所以每的面的次数至少为3. 设每个面次数至少为x，则

$$\begin{cases} 10x \le 30 \\ x \ge 3 \end{cases} \Rightarrow x=3.$$

又 $3\times 10=30$，所以每个面的次数即为3.

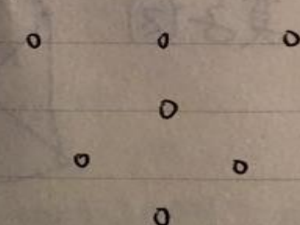
所以该连通平面图是极大平面图。

附加题：证明下图不是哈密顿图。



证明：取 $V_1=\{a,b,c,d,f,g\}$

$G-V_1$ 如下图所示



$p(G-V_1)=7>6$，故不存在哈密顿

组合分析

第8 章 组合分析初步

- 8.1 加法法则与乘法法则
- 8.2 基本排列组合的计数方法
- 8.3 递推方程的求解与应用

8.1 加法法则和乘法法则

- 加法法则与乘法法则
- 应用实例

加法法则

事件 A有 m 种产生方式，事件 B 有 n 种产生方式，则“事件 A 或 B”有 $m+n$ 种产生方式.

使用条件：事件 A与 B 产生方式不重叠

适用问题：分类选取. 方式分别计数，再相加.

推广：事件 A_1 有 n_1 种产生方式，事件 A_2有 n_2 种产生方式，…, 事件 A_k 有 n_k 种产生的方式，则“事件A_1 或 A_2或…A_k” 有 $n_1+n_2+\ldots+n_k$ 种产生的方式.

乘法法则

事件 A有 m 种产生方式，事件 B 有 n 种产生方式，则“事件 A 与 B”有 mn 种产生方式.

使用条件：事件A与B产生方式相互独立

适用问题：分步选取. 方式是连续的步骤，各步相互独立，分别计数，然后相乘.

推广：事件 A_1 有 n_1 种产生方式，事件 A_2有 n_2 种产生方式，…, 事件 A_k 有 n_k 种产生的方式，则“事件A_1 与 A_2与 …A_k” 有 $n_1n_2\ldots n_k$ 种产生的方式.

应用实例

例1　由数字 1、2、3、4、5 构成 3 位数.

(1) 如果各位数字都不相同，那么有多少种方法?

(2) 如果必须是偶数，则有多少种方法?

(3) 其中可以被 5 整除的有多少个?

(4) 其中比300大的有多少个?

解　(1)　$5\times4\times3=60$.

(2)　个位为2,4，十位、百位各5种：　$2\times5\times5=50$.

(3)　个位为5，十位和百位同(2)：　$1\times5\times5=25$.

(4)　百位取3,4或5，十位和个位各5种:

$$3\times5\times5=75.$$

应用实例

例2 求 1400 的不同的正因子个数

解 $1400 = 2^3\, 5^2\, 7$

正因子为：$2^i\, 5^j\, 7^k$,

$0 \le i \le 3$， $0 \le j \le 2$， $0 \le k \le 1$

$N = (3+1)(2+1)(1+1) = 24$

8.2 基本排列组合的计数方法

- 排列组合的分类
- 集合的排列
- 集合的组合
- 多重集的排列
- 多重集的组合

排列与组合

- 排列
  - 5天内安排3门考试，每天只考一门，有多少种安排?
  - 从1~9中选取数字能构成多少个四位数?
- 组合
  - 从5种不同的球中每次取3个不同的球，有多少种方法?
  - 从5种不同的球中每次取3个球，有多少种方法?

排列组合的分类

选取问题：设 n 元集合 S，从 S 中选取 r 个元素. 根据是否有序，是否允许重复可将该问题分为四个子类型

	不重复	重复
有序	集合排列 $P(n,r)$	多重集排列
无序	集合组合 $C(n,r)$	多重集组合

集合的排列

从 n 元集 S 中有序、不重复选取的 r 个元素称为 S 的一个 r 排列，S 的所有 r 排列的数目记作 P_n^r 或P(n,r). 如果r=n，则称为S的全排列，简称为S的排列

$$P_n^r = \begin{cases} \dfrac{n!}{(n-r)!} = \dfrac{n!}{(n-r)!} & n \geq r \\ 0 & n < r \end{cases}$$

S 的 r- 环排列 数 $= \dfrac{P_n^r}{r} = \dfrac{n!}{r(n-r)!}$

实例

例4 排列 26个字母，使得a与b之间恰有7个字母，求方法数.

解：固定a 和 b 中间选7个字母，有 $2P_{24}^{7}$ 种方法

将它看作大字母与其余 17个全排列有18！种，

$$N = 2P_{24}^{7} \cdot 18!$$

实例（续）

例5

(1) 10个男孩，5个女孩站成一排，若没女孩相邻，有多少种方法？

(2) 如果站成一个圆圈，有多少种方法？

解:

(1) $P_{10}^{10} P_{11}^{5}$

(2) $\dfrac{1}{10} P_{10}^{10} P_{10}^{5}$

集合的组合

- 从 n 元集 S 中无序、不重复选取的 r 个元素称为S的一个r 组合，S 的所有 r 组合的数目记作 C_n^r 或者C(n,r)或者$\binom{n}{r}$

$$C_n^r = \begin{cases} \dfrac{P_n^r}{r!} = \dfrac{n!}{r!(n-r)!} & n \geq r \\ 0 & n < r \end{cases}$$

推论

- 对一切r≤n有

$$C_n^r = C_n^{n-r}$$

证明方法:

公式代入

组合证明（一一对应）

基本计数公式的应用

例1 从1—300中任取3个数使得其和能被3整除有多少种方法?

解 令 $A=\{1, 4, \ldots, 298\}$，$B=\{2, 5, \ldots, 299\}$

$C=\{3, 6, \ldots, 300\}$

将方法分类:

分别取自 A, B, C: 各 C_{100}^{3}

A, B, C各取1个: C_{100}^{1}

$$N = 3C_{100}^{3} + (C_{100}^{1})^{3} = 1485100$$

基本计数公式的应用（续）

例2 求1000！的末尾有多少个0?

解 $1000! = 1000 \times 999 \times 998 \times \ldots \times 2 \times 1$

将上面的每个因子分解，若分解式中共有 i 个5, j 个2，那么 $\min\{ i, j \}$ 就是 0 的个数.

1, …, 1000 中有

500 个是 2 的倍数，$j > 500$;

200 个是 5 的倍数，

40 个是 25 的倍数（多加40个5），

8 个是 125 的倍数（再多加8个5），

1 个是 625 的倍数（再多加1个5）

$i = 200 + 40 + 8 + 1 = 249$. $\min\{ i, j \}=249$.

多重集的排列

多重集 $S = \{n_1 \cdot a_1, n_2 \cdot a_2, \ldots, n_k \cdot a_k\}$，$0 < n_i \le +\infty$

(1) 全排列 $r = n, n_1 + n_2 + \ldots + n_k = n$

$$N = \frac{n!}{n_1! n_2! \ldots n_k!}$$ 可简记为 $$\binom{n}{n_1 \quad n_2 \quad \ldots \quad n_k}$$

证明：分步选取，先放 a_1, 有 $C_n^{n_1}$种方法；再放 a_2, 有$C_{n-n_1}^{n_2}$种方法，... , 放a_k有 $C_{n-n_1-n_2-\ldots-n_{k-1}}^{n_k}$ 种方法

$$N = C_n^{n_1} C_{n-n_1}^{n_2} \ldots C_{n-n_1-\ldots n_{k-1}}^{n_k} = \frac{n!}{n_1! n_2! \ldots n_k!}$$

(2) 若 $r \le n_i$时，每个位置都有 k 种选法，得 k^r.

实例

- 把**2**面红旗**3**面黄旗依次悬挂在一根旗杆上，可组成多少种不同的标志？

解　本题相当于求多重集**{2** •红旗，3 •黄旗**}**的排列数**N**，由定理可知

$$N = \frac{5!}{2! \times 3!} = 10$$

实例

- 有**10**种画册，每种数量不限，取**3**本送给**3**位朋友，有多少种方法?

解 问题相当于多重集$\{\infty \bullet a_1, \infty \bullet a_2, \ldots, \infty \bullet a_{10}\}$的**3**排列，由定理可知$N=10^3=1000$

小结——多重集排列问题

- r>n，有N=0
- r=n，有 $N=\dfrac{n!}{n_1!n_2!\ldots n_k!}$
- r<n，且对一切i=1,2,…,k有$n_i \geq r$，则$N=k^r$
- r<n，且存在$n_i<r$，则对N没有一般的求解公式，可以使用其他的组合数学方法解决

多重集的组合

当$r \le n_i$， 多重集 $S = \{ n_1 \cdot a_1, n_2 \cdot a_2, \ldots, n_k \cdot a_k \}$ 的组合数为 $N = \dfrac{(r+k-1)!}{r!(k-1)!} = C_{k+r-1}^{r}$

证明 一个 r 组合为 $\{ x_1 \cdot a_1, x_2 \cdot a_2, \ldots, x_k \cdot a_k \}$，其中 $x_1 + x_2 + \ldots + x_k = r$，$x_i$ 为非负整数. 这个不定方程的非负整数解对应于下述排列

$$\underbrace{1\ldots1}_{x_1\text{个}}\ 0\ \underbrace{1\ldots1}_{x_2\text{个}}\ 0\ \underbrace{1\ldots1}_{x_3\text{个}}\ 0\ \ldots\ldots\ 0\ \underbrace{1\ldots1}_{x_k\text{个}}$$

r 个1，k-1个 0 的全排列数为 $N = \dfrac{(r+k-1)!}{r!(k-1)!} = C_{k+r-1}^{r}$

推论

当$n_i \ge r \ge k$, 从多重集 S ={ $n_1 \cdot a_1$, $n_2 \cdot a_2$, …, $n_k \cdot a_k$ }中每个元素至少取一个的r组合数为 

证明　任取一个所求的r组合，从中拿走元素 $a_1, a_2, \ldots, a_k$ ，就得到S的一个(r-k)组合（或者反之）。

实例

例3 r 个相同的球放到 n 个不同的盒子里，每个盒子球数不限，求放球方法数.

解：设盒子的球数依次记为 $x_1, x_2, \ldots, x_n$, 则满足下述方程：

$x_1 + x_2 + \ldots + x_n = r$， $x_1,x_2, \ldots, x_n$为非负整数

该方程的解的个数为：

$$N = \frac{(r+n-1)!}{r!(n-1)!} = C_{n+r-1}^{r}$$

实例（续）

例6 把 $2n$ 个人分成 n 组，每组2人，有多少分法？

解：相当于 $2n$ 不同的球放到 n 个相同的盒子，每个盒子 2个，放法为

$$N=\frac{(2n)!}{(2!)^n n!}=\frac{(2n)!}{2^n n!}$$

实例（续）

例7 9本不同的书，其中4本红皮，5本白皮.

(1) 9本书的排列方式数有多少?

(2) 若白皮书必须放在一起，那么有多少方法?

(3) 若白皮书必须放在一起，红皮书也必须放在一起，那么有多少方法?

(4) 若白皮和红皮书必须相间，有多少方法?

解:

(1) 9!

(2) 5！5！

(3) 5！4！2！

(4) 5！4！

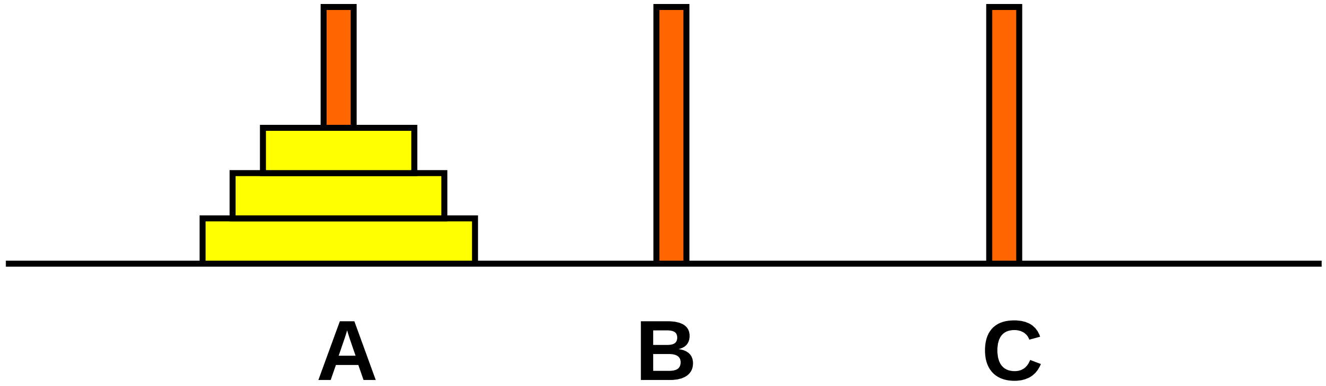
8.3 递推方程的求解与应用

- Hanoi 塔问题
- 递推方程的定义
- 二分归并排序算法的分析
- 快速排序算法的分析
- 递归树
- 分治算法分析的一般公式

Hanoi塔问题

Hanoi塔问题:

从*A*柱将这些圆盘移到*C*柱上去. 如果把一个圆盘从一个柱子移到另一个柱子称作 1 次移动，在移动和放置时允许使用*B*柱，但不允许大圆盘放到小圆盘的上面. 问把所有的圆盘的从*A*移到*C*总计需要多少次移动?



算法 Hanoi (A,C,n)　//*把n个盘子从A移到C

1. Hanoi (A,B,n-1)

2. move (A,C)　　　//*把1个盘子从A移到C

3. Hanoi (B,C,n-1)

移动n个盘子的总次数为$T(n)$，得到递推方程

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

$$T(1)=1.$$

可以求得 $T(n)=2^n-1$

1秒钟移动1次，64个盘子大约需要5000亿年

$$
\begin{aligned}
T(n) &= 2T(n-1)+1 \\
&= 2[2T(n-2)+1]+1 \quad (T(n-1)\text{被含}T(n-2)\text{的项替换}) \\
&= 2^2T(n-2)+2+1 \\
&= 2^2[2T(n-3)+1]+2+1 \quad (T(n-2)\text{被含}T(n-3)\text{的项替换}) \\
&= 2^3T(n-3)+2^2+2+1 \\
&= \ldots \\
&= 2^{n-1}T(1)+2^{n-2}+2^{n-3}+\ldots+2+1 \\
&= 2^{n-1}+2^{n-2}+2^{n-3}+\ldots+2+1 \quad (\text{代入初值}) \\
&= 2^n-1 \qquad\qquad \text{（等比级数求和）}
\end{aligned}
$$

递推方程的定义

定义10.5 设序列$a_0, a_1, \ldots, a_n, \ldots$, 简记为$\{a_n\}$, 一个把$a_n$与某些个$a_i$（$i<n$）联系起来的等式叫做关于序列$\{a_n\}$的递推方程.

实例：

Fibonacci数列： $f_n=f_{n-1}+f_{n-2}$, 初值 $f_0=1$, $f_1=1$

阶乘数列$\{a_n\}$，$a_n=n!$：$a_n=na_{n-1}$, $a_1=1$

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

递推方程的求解方法——迭代法

- 从原始递推方程开始
- 把表达式中的后项用相等的前项的表达式代入（利用方程所表达的数列中后项对前项的依赖关系）
- 直到表达式中没有函数项为止
- 将右边的项求和并将结果进行化简
- 为了保证结果的正确性，往往需要带入原来的递推方程进行验证

二分归并排序算法

算法**Mergesort**(A,s,t) //*排序数组$A[s..t]$

1. $m \leftarrow (t-s)/2$
2. $A \leftarrow$ **Mergesort**(A,s,m) //*排序前半数组
3. $B \leftarrow$ **Mergesort**($A,m+1,t$) //*排序后半数组
4. **Merge**(A,B) //*将排好序的A,B归并

假设$n=2^k$，比较次数至多为$W(n)$

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

归并两个$n/2$大小数组的比较次数至多为$n-1$

实例

输入：[5, 1, 7, 8, 2, 4, 6, 3]

划分：[5, 1, 7, 8], [2, 4, 6, 3]

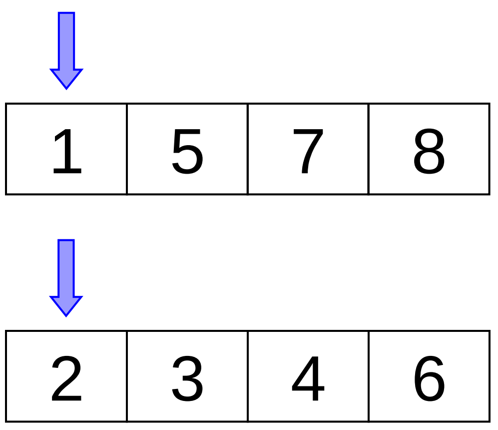
递归排序前半个数组: [5, 1, 7, 8] ⇒ [1, 5, 7, 8]

递归排序后半个数组: [2 ,4, 6, 3] ⇒ [2, 3, 4, 6]

归并: [1, 5, 7, 8] 和 [2, 3, 4, 6]

输出：[1, 2, 3, 4, 5, 6, 7, 8]

归并过程



求解递推方程

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

$$
\begin{aligned}
W(n) &= 2W(2^{k-1}) + 2^k - 1 \\
&= 2[2W(2^{k-2}) + 2^{k-1} - 1] + 2^k - 1 \\
&= 2^2 W(2^{k-2}) + 2^k - 2 + 2^k - 1 \\
&= 2^2[2W(2^{k-3}) + 2^{k-2} - 1] + 2^k - 2 + 2^k - 1 \\
&= 2^3 W(2^{k-3}) + 2^k - 2^2 + 2^k - 2 + 2^k - 1 \\
&= \ldots \\
&= 2^k W(1) + k2^k - (2^{k-1} + 2^{k-2} + \ldots + 2 + 1) \\
&= k2^k - 2^k + 1 \\
&= n\log n - n + 1
\end{aligned}
$$

归纳法验证解

$$n\log n - n + 1$$

- n=1代入上述公式得

$$W(1)=1\log 1-1+1=0,$$

符合初始条件.

- 假设对于任何小于n的正整数t，$W(t)$都是正确的，将结果代入原递推方程的右边得

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

$$=2(2^{k-1}\log 2^{k-1}-2^{k-1}+1)+2^k-1$$

$$=2^k(k-1)-2^k+2+2^k-1=k2^k-2^k+1$$

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

快速排序算法

算法 Quicksort(A,p,r) //*排序数组$A[p..r]$

输入：数组$A[p..r]$

输出：排好序的数组A

1. if $p < r$
2. then $q \leftarrow$Partition(A, p, r) //*以$A[p]$为准划分A
3. $A[p] \leftrightarrow A[q]$ //*$A[p]$与$A[q]$交换
4. Quicksort(A,p,q-1) //*对子数组递归排序
5. Quicksort(A,q+1$,r$)

划分过程

```
Partition(A,p,r)
  1.  x ← A[p]
  2.  i ← p
  3.  j ← r+1
  4.  while true do
  5.      repeat j ← j –1
  6.      until A[ j ]<x  //*右边第1个比A[p]小的A[j]
  7.      repeat i ←i +1
  8.      until A[ i ]>x  //*左边第1个比A[p]大的A[i]
  9.      if i < j
  10.         then A[ i ] ↔ A[ j ]  //*交换A[j]与A[i]
  11.         else return j
```

27	99	0	8	13	64	86	16	7	10	88	25	90

27	25	0	8	13	64	86	16	7	10	88	99	90

27	25	0	8	13	10	86	16	7	64	88	99	90

27	25	0	8	13	10	7	16	86	64	88	99	90

16	25	0	8	13	10	7	27	86	64	88	99	90

平均时间复杂度

- $T(n)$为对数组的各种输入平均做的比较次数
  将输入按照$A[p]$在排好序后的位置分别为1, 2, …, n进行分类. 假设每类输入出现的概率相等
- $A[p]$处位置1，划分后子问题规模分别为0和n-1
  …
  $A[p]$处位置n，划分后子问题规模分别为n-1和0
- n 种输入的平均复杂度为

$$T(n) = \frac{1}{n}[(T(0)+T(n-1))+(T(1)+T(n-2))+\ldots$$

$$+(T(n-1)+T(0))]+O(n) = \frac{2}{n}\sum_{i=1}^{n-1} T(i)+O(n)$$

递推方程求解

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

差消法化简

$$nT(n)=2\sum_{i=1}^{n-1}T(i)+cn^2 \quad c\text{为某个常数}$$

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

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

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

迭代

$$\frac{T(n)}{n+1}=\frac{T(n-1)}{n}+\frac{c}{n+1}$$ c为某个常数

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

$$=\cdots$$

$$=c[\frac{1}{n+1}+\frac{1}{n}+\ldots+\frac{1}{3}+\frac{T(1)}{2}]$$

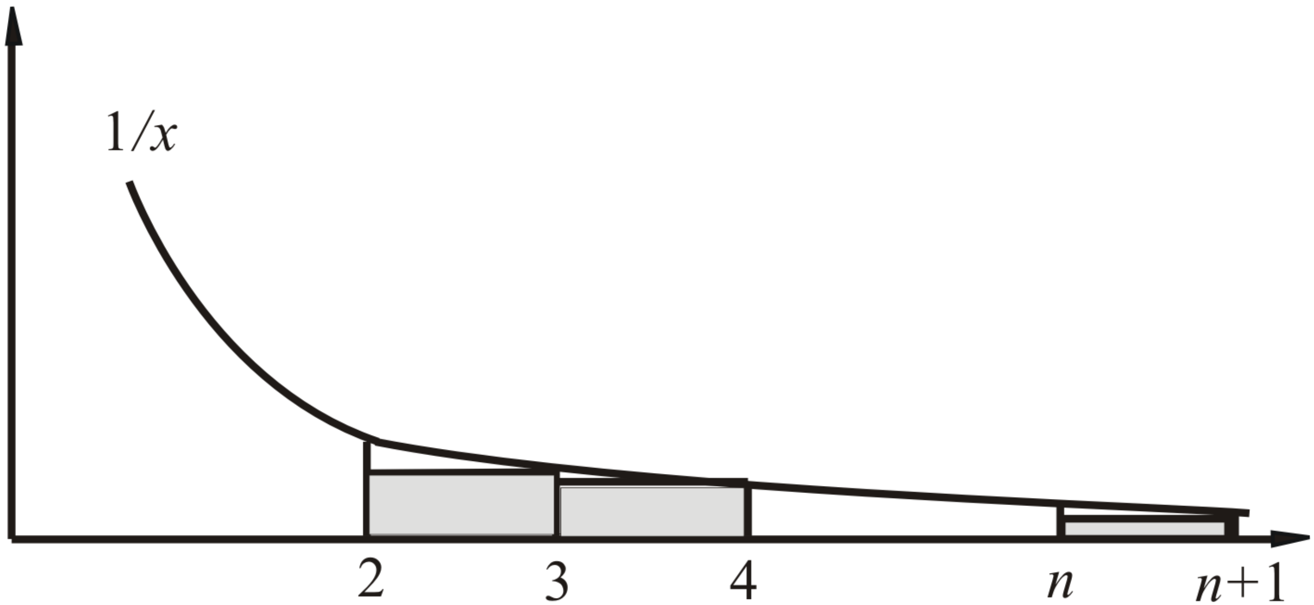
$$=c\left[\frac{1}{n+1}+\frac{1}{n}+\ldots+\frac{1}{3}\right]$$

用积分近似

$$\frac{1}{n+1}+\frac{1}{n}+\ldots+\frac{1}{3}\le\int_{2}^{n+1}\frac{1}{x}dx=\ln x\Big|_{2}^{n+1}$$

$$=\ln(n+1)-\ln 2=O(\log n)$$

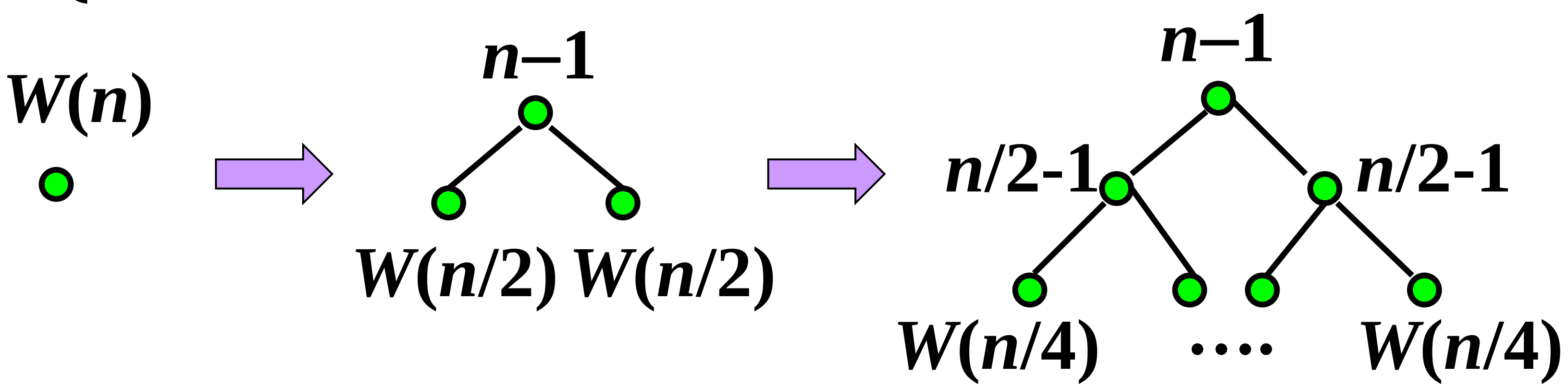
$$T(n)=O(n\log n)$$



递归树

带权二叉树，每个结点有权. 初始只有一个结点，权为W(n). 然后不断迭代，将函数结点用与其相等的递推方程右部的子树替代，直到树中不再含有权为函数的结点.

$$\begin{cases} W(n) = 2W(n/2) + n - 1, \; n = 2^k \\ W(1) = 0 \end{cases}$$



n-1

n/2-1 n/2-1

n/4-1 n/4-1 n/4-1 n/4-1

……..

1 1 1 1 1 ……….. 1 1 1 1 1

n-1

n-2

n-4

n-2^{k-1}

$$(n-1)+(n-2)+\ldots+(n-2^{k-1})$$

$$=nk-(1+2+\ldots+2^{k-1})$$

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

分治算法

- 算法设计中的一种重要技术
- 主要思想：
  - 将原问题归约成规模更小的子问题
  - 用同样的算法递归地求解每个子问题
  - 将子问题的解进行综合，从而得到原问题的解

分治算法的常用递推公式

$$\begin{cases} T(n) = aT(n/b) + d(n) & n = b^k \\ T(1) = 1 \end{cases}$$

其中*a*为子问题个数，*n/b*为子问题规模，*d(n)*为分解成子问题或组合解的代价

方程的解为：

$$d(n) = c: \quad T(n) = \begin{cases} O(n^{\log_b a}) & a \neq 1 \\ O(\log n) & a = 1 \end{cases}$$

$$d(n) = cn: \; T(n) = \begin{cases} O(n) & a < b \\ O(n\log n) & a = b \\ O(n^{\log_b a}) & a > b \end{cases}$$

迭代

$$\begin{cases} T(n) = aT(n/b) + d(n) & n = b^k \\ T(1) = 1 \end{cases}$$

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

$$= \ldots$$

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

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

Case1 $d(n)=c$

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

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

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

Case2 $d(n)=cn$

$$n = b^k$$

$$T(n) = a^k + \sum_{i=0}^{k-1} a^i \frac{cn}{b^i} = a^k + cn\sum_{i=0}^{k-1}\left(\frac{a}{b}\right)^i$$

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

应用实例

$$d(n)=cn:\ T(n)=\begin{cases} O(n) & a<b \\ O(n\log n) & a=b \\ O(n^{\log_b a}) & a>b \end{cases}$$

- 二分归并

$$\begin{cases} W(n)=2W(n/2)+n-1,\ n=2^k \\ W(1)=0 \end{cases}$$

a=2，b=2，$d(n)$=$O(n)$ $\Rightarrow$ $T(n)$=$O(n\log n)$

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

- 二分查找

$$\begin{cases} W(n)=W(n/2)+1,\ n=2^k \\ W(1)=0 \end{cases}$$

a=1，b=2，$d(n)$=$O(1)$ $\Rightarrow$ $T(n)$=$O(\log n)$

例题：关系计数

设A为n元集，A上可定义多少个不同的二元关系，其中有

(1) 多少个自反的二元关系?

(2) 多少个对称的二元关系?

(3) 多少个反对称的二元关系?

解：可定义二元关系个数: 2^{n^2} ，其中

(1) 自反关系个数： 2^{n^2-n}

(2) 对称关系个数： $2^{\frac{n(n+1)}{2}}$

(3) 反对称关系个数： $2^n 3^{\frac{n(n-1)}{2}}$

例题：函数计数

设*A*、*B*分别为*m*元集和*n*元集，*m*和*n*为正整数，则从*A*到*B*有多少个函数？

(1) 当*m*与*n*满足什么条件时存在单射函数？有多少个?

(2) 当*m*与*n*满足什么条件时存在双射函数？有多少个?

有n^m个从*A*到*B*的函数，其中

(1) 当$m \le n$时存在单射函数. 单射函数有 P_n^m 个

(2) 当$m=n$时存在双射函数. 双射函数有$n!$个.

P_n^m

作业（详细写出算式得出过程）

1、已知从1到n的十进制正整数的总数字个数（不包括无效0）是1890，求n

2、三只白色的棋子和两只红色的棋子摆放在5X5棋盘上，要求每行每列只放置一个棋子，则共有多少种不同的摆放方法?

3、有6个演唱节目，4个舞蹈节目，要编节目单，要求任意两个舞蹈节目之间至少安排一个演唱节目，则共可编写出多少个节目单?

作业

- 课后题8.27, 8.28, 8.31, 8.32