1.1 逻辑

- 命题是什么
- 蕴含
- 逻辑运算符的优先级
- 将命题用逻辑符号表达

1.2 逻辑等值式

- 重言式和矛盾式

  并要记住常见的式子

- 用真值表证明等值式

- 用已知的逻辑等值式证明等值式

1.3 谓词与量词

- 谓词、量词
- 辖域、变量的自由性

1.4 嵌套量词

- 用含有嵌套量词的语句将命题符号化，翻译成逻辑语言
- 含有量词的命题的False情况
- 量词的顺序改变语义

注意书写的格式

1.5 推理规则

- 推理的公式，比如附加、合取、假言推理等

- 从已知条件通过推理规则能够得到结论，等价于将所有前提合取后，推出结论命题是重言式

- 推理证明

- 存在量词的推理证明，记住四个定理进行量词的删增

  UG, UI, EI, EG

  删去量词后用推理规则证明即可

- 空证明与简单证明

  空证明是证明前件为假，简单证明是证明后件为真

1-Paradox 悖论

- 悖论

  设一个命题为B，由B经过一系列推理可以得到非B，由非B经过一系列推理也可得到B

2.1 集合

- 集合相等的条件：元素相等

  重复列举是没有用的

- 幂集

  P（S）

- 基数，用绝对值符号表示
- 笛卡尔积

- 超集：包含A、B的集合。A并B是最小的超集
- A交B是最小的A、B共同元素子集
- 差集

  A - B又可以称为：B相对A的补

  规则是：A减去一部分元素，该部分元素在B中也存在

- $\bar{A}$ 表示A的补集，注意题目中全集是什么
- 集合恒等的证明
  - 证明互为子集

    证明一个是另一个的子集时，可以列出前件的意义（注意"或"要分类讨论），也即x∈Q，然后推x也属于后件

  - 利用集合的命题表示 + 逻辑恒等式

    也即将要证明的东西翻译成命题逻辑，证明完成后再翻译回集合

  - 利用成员表

    和真值表类似，也即假定一个元素，然后列出它和集合A、B的关系，往下递推得到结论

- n元并、交，generalize-泛化，可以用大U或者'Big Arch'表示
- 计算机中的并交，其实就是按位与、或
- 集合计数

  单数加偶数减，均为交集

2.2 函数

- 定义

  $A \to B$ 的函数，元素是A的全集，值域是B的子集，B叫做陪域

  函数f可以看作一个关系，那么就有(a,b)∈f

  Y^X 表示$X \to Y$ 所有可能的函数，有$|Y|^{|X|}$个

- 函数的像，原像可以是单个元素，也可以是定义域的子集，得到的像相应的是一个集合，不过可能只有一个元素

- 单射：每个像只有一个原像

  也即值域和定义域有相同的基数

  但陪域可能大于值域

- 单调函数一定单射

- 满射（Onto）：值域等于陪域

- 恒等函数，像等于原像

- 双射（*one-to-one* **correspondence**， bijective）

  只有双射函数存在反函数

  反函数和原函数的符合为恒等函数

- 函数运算符

  复合：课内介绍的是右复合，也即 **f圈g等于f(g(a))**

3.1 关系及其性质

- 逆关系

- 关系何时可以看作函数

  total：用到了A中所有元素，否则称为partial

  functional：对于用到的A中的元素，存在有且仅有一个B中的值与其对应

- 基本是讨论在单个集合上的关系

- 关系的性质

  - 自反性、反自反性

    自反是所有恒等元素都要在

    反自反是所有恒等元素都要不在

    - 定理：一个集合是反自反的当且仅当它的补关系是自反的

    小于等于、包含、整除关系都是自反的

  - 对称性、反对称性

    对称是只要存在（a，b），就要有（b，a）

    反对称是存在（a，b），就不可存在（b，a），除非a = b

    - 相当于均不关注恒等元素

  - 传递性

- 复合关系

书上介绍的是左复合，也即：

若 R：A → B， S：B → C， 那么$S \circ R$ 是表示 A → C

其实想一下左右有讲究的情况，上一次应该是在矩阵，左乘与右乘

- 定理1

  如果在A上的关系R是传递的，那么$R^n \subset R$

- 关系的n次方，左复合

3.2 n元关系

- 由多个集合的笛卡尔积的子集构成

  多个集合称为关系的定义域

  集合的数量称为关系的度

- 关系型数据库

  键与反键

  - 选择操作符，其实也就是对已存储的关系，找出符合要求的对象，提取出一整项信息

  - 项目操作符，将关系的某几项取出

  - 合并操作符

    对于一个新的关系R2，将其与R1合并的操作是根据主键进行信息添加

    那么字典的键可以看作主键

    对于python，键可以是一个元组，那么要寻找键值的话，需要多个键的信息同时匹配，比如说一个年级的学生学号，可以是(id, grade)比如('id', ''2023')

- SQL语句

3.3 关系的表示

- n元组的列表

  一个n元组的函数，映射到 T, F 也即0，1

- 对于二元关系

  可以用0-1矩阵或者有向图表示

  - 0 - 1 矩阵便于看出关系的性质
  - 有向图构成了后面一系列课程的概念，顶点、边、起终点

  边集是在点集上的关系

  有向图查看关系：

  - 自反：每个顶点都有圈

    - 对称：两个顶点A、B，若有边，则是双向边
    - 传递：如果A、C是可达的，那么A、C之间有直接相连的路
- 0 - 1 矩阵对于关系的运算非常方便

  关系的并、交分别对于矩阵并、交运算

  关系的传递对于矩阵乘法

  n次方是R的矩阵右乘 R^{n-1} 的矩阵

3.4 关系的闭包

- 对于关系R的某一性质X，R的X闭包是指，具有X性质的R的最小超集
    - 自反闭包

    - 对称闭包

    - 传递闭包

      引入$R^* = \cup_{i=1}^{n} R_i$

      此即R的传递闭包，包含了所有可达路径

        - 定理：如果R是传递的，R的n次方仍为R的子集
- 引理1

  如果R是n元集合A上的关系，若关系中至少存在一条可达路径a → b，那么R中存在一条长度不超过n的路径

  特别地，如果a ≠ b，那么存在一条长度不超过n-1的路径
- 定理3

  传递闭包可以用矩阵运算实现
- Worshell 算法

  计算传递闭包的最快算法

  W^k 表示对第k个对角线元素处理完后的结果

3.5 等价关系

- 定义：R是一个等价关系 $\leftrightarrow$ R是自反、对称、传递的
- 一些常见的等价关系
    - 同模
    - 相等
- 等价类
    - 定义：在一个等价关系R中，与a相关的元素（也即aRb）构成的集合
    - 表示，如果整个讨论范围内只有一个关系，可以把R去掉

    - 任选等价类中的一个元素都可以来表示这个等价类整体，因此说这个等价关系是loose的（松动的）
- 划分、割集
    - 定义：A的不相交子集，并且这组子集的并集为A

      也即把A分成几个部分，没有剩余元素
    - 集合A的割集可以通过在集合A上的等价关系得到，每个等价类就是一个不相交子集
    - 那么一个划分其实和在A上的一个等价关系是一 一对应的

3.6 偏序关系

- 偏序关系：自反、反对称、传递
- 常见的偏序关系:
    - 大于等于关系
    - 整除关系
    - 包含关系
- 如果两个属于集合的元素在集合上的偏序关系中，称这两个元素是可比的
- 如果集合S中的任意两个元素都是可比的，称S为全序集或者线性序列集合

  此时偏序称为全序或线序

  全序集又称为链
- 良序集：偏序集是全序的，并且S的非空子集都有最小值

  一个是全序集而不是良序集的例子：实数域上的小于或大于关系
- 字典顺序：
- 汉斯图
- 极大、极小元

  可视化地来说，也就是汉斯图的底部、顶部，而最大、最小元为唯一顶部、底部
- 偏序集的子集的上、下界，upper/lower bound

  作为界必须在该子集元素的上、下方，且在图中“不走回头路”，如果上界就要找往上的共同路径

  上、下界可以为该子集的元素
- 最小上界、最大下界

  必须满足与其它上界 / 下界是可比的并且 小于 / 大于其它任意上界 / 下界

  不可等于，也就是只存在一个，如下图，b和c是没有共同的最小上界的，在它们的上界edf中，ed不构成可比关系
- 格（lattice）

  在集合A上的偏序关系S，若任意两个A中元素都有着相同的最小上界和最大下界，那么该偏序集合称为格

- 拓扑排序

  数据结构中也会介绍到，是一个关于图论的算法吧，DFS、Kahn算法都实现了拓扑排序

  对于离散数学，首先介绍一个引理：

  > 对于非空的偏序集，总存在一个极小元

  那么按顺序，每次取出极小元构成的数组便是拓扑排序的结果（存在多个极小元时任选一个均可）

  对于偏序关系中不连通的部分，各自进行拓扑排序后合并即可

4.1 图的简介

- 图的概念：

  由非空的点集和边集组成

- 多重图：

  具有平行边，也即起终点相同的边

- 伪图：

  具有圈，也可具有平行边

- 有向图，也即边有方向，是有序对
- 有向多重图，可以有圈和平行边

4.2 图的术语

- 出度为正，入度为负
- 一些常见的简单图
  - 完全图，每个顶点都与其它顶点邻接
  - 圈，也即顺序相连
  - 轮，在圈的基础上加入了一个与原先所有顶点都邻接的顶点
  - n元立方体，用一个比特串去表示，当两个顶点的比特串有且仅有一位不同时，它们是邻接的

    可以由n元立方体非常轻易的得到n + 1元立方体，，只需要在原先基础上，在前面增加一位，分别为 0 和 1开头即可，再把
- 二分图
  - 定义：

    当且仅当其顶点集可以分成两个独立的集合（通常称为两个“部”），使得图中任何一条边都只连接这两个集合中的顶点，即同一集合中的顶点之间没有边相连
  - 任何偶数个顶点的环图都是二分图，因为我们可以交替地将相邻顶点分配到两个集合中

    比如C6的环图，可以将顶点1，3，5分在一组，2，4，6分在一组便得到了一个二分图的划分

    那么有一个技巧，其实先选择一个顶点入A组，其邻接入B组，对A、B组所在的元素，利用一个队列去维护，逐一的去判断（其实就是BFS算法）

- 完全二分图

  除了要满足能划分成谔谔分图的条件，还要满足划分A中的任一顶点要与划分B中的所有顶点都邻接

- 并行算法
- 子图

  G的子图G'，其中 V‘ 和 E' 分别是 V 和 E 的子集

- 图的并运算，将所有的边集、点集并在一起

4.3 图的表示和同构

- 邻接表

  一列表示顶点，同行列出其所有邻接的边

  对于有向图、稀疏图更加高效

  左侧是起点，右侧是其邻接的终点

- 邻接矩阵

  其实这里在回顾一下边的定义，它就只是一个二元组，对于有向图，是一个有序二元组，仅此而已

  - 一些邻接矩阵的性质
    - 对于无向图来说，邻接矩阵对称
    - 如果有圈，对角线元素为1，简单图对角线元素均为0
  - 邻接矩阵也可以表示多重图，此时矩阵元素代表了多重边的数量
  - 有向图的边是由有序对构成的，其邻接矩阵具有顺序，一般（a，b），a对应行，b对应列

- 关联矩阵
  - no！关联矩阵的行列意义要特别注意，其实它的意思是某条边跟哪两个顶点关联
  - 行是顶点，列是边，每列只有两个元素是非0的

    对于无向图而言如果两个顶点有路可达，那么该位置就为1

    对于有向图而言，起点为1，终点为 负1

    0均表示不关联

- 同构图
  - 定义：有一个一 一对应的函数将点集V1映射到点集V2，保证映射前后相应元素的邻接关系不变。

    其实就是将点重命名罢了，重命名之后两个图如果是能完全一样的，那么就是同构的，这就是informal 的 definition

  - 同构的图，其性质也是完全相同的，列举一些重要的
    - 结点数和边数
    - 出入度数
    - 是否为二分图

  - 判断同构的算法，已知的算法里，最坏的情况是指数级别的时间复杂度

    但实际上，很少有这样的图
  - 一个必要但不充分的同构判定
    - 点、边数相等
    - 每个顶点的出入度相等
    - 两个图，其中一个图的子图在另外一个图也要能找到（那其实提供了一种根据形状来判定的方法）
  - 如果同构的话，给出映射关系
  - 可以用邻接矩阵来判断同构关系或者给出映射关系，只需要相同列即可
  - 有一个NAUTY算法，在一秒内可以计算出两个100顶点内的图是否同构
    - 如果是做题判断是否同构的话，可以先进行匹配，出入度相同的顶点

  暂时没有判断两个图是否同构的充分条件

4.4 图的连通性

- 不含重复边的路或者回路是简单路
- 连通是指任意两个顶点都是可达的
- 任意一个图都可以由多个连通的不相交子图的并集组成，这些连通的子图叫做图的连通分量
- 割点、割边
- 强连通：有向图中任意两点（有序对）均存在可达路径

  弱连通：看作无向图时连通
- 最大强连通子图
- 一个特定长度的简单回路可以判定两个图是否同构，并给出映射关系

  判断不是同构，如下

  判断是同构，如下

  给出一个图的一条回路，写出各位置的度数，在另外一个图中根据这个度数找一条回路即可

- 两个结点间的路径长为r的路径条数

  利用邻接矩阵的r次方便可以给出
- 应用，解题：

  可以用一个二分图来表示某人是否具有某项性质

4.5 欧拉图和汉密尔顿图

- 欧拉图

  边遍历，不重复

  判定无向图有欧拉回路的充分必要条件！
  - 每个结点都必须为偶度数

- 图为连通多重图

判定有向图有欧拉回路的充要条件

- 每个结点出入度相等
- 图是强连通的

如果是判断有向图是否有欧拉路：

- **图中恰好有两个顶点的入度和出度差为1**，其余顶点的入度和出度相等

  出度多1的为起点，入度多1的为终点

- 图是强连通的

无向图欧拉路的非回路

- **有且只有两个顶点的度数是奇数**，其余顶点的度数是偶数。

  起点终点为这两个奇度数结点

- 图是 **连通的**，即从任意一个顶点出发，都能通过边到达图中的所有其他顶点。

- 欧拉回图的构造

  可以分治，找一个公共顶点处理，将两条回路合并即可，直至没有边重复

  称为Hierholzer算法，必须先判断满足充要条件，然后步骤如下

  1. 选择任一顶点为起点，遍历所有相邻边。
  2. 深度搜索，访问相邻顶点。将经过的边都删除。
  3. 如果当前顶点没有相邻边，则将顶点入栈。
  4. 栈中的顶点倒序输出，就是从起点出发的欧拉回路。

  对于书上的例子，简单一些来说，就是将大图逐渐小化，每次得到一个子图，再对原图删除该子图的部分继续操作，找合适的相接点即可

- Hamilton图

  点遍历，无须经过所有边，每个点只能经过一次

  没有判定Hamilton回路的充要条件

  不过有一些判断不是充分条件：

  - 图中若某个点度数为1，不可能有汉密尔顿回路
  - 汉密尔顿回路内不可能有一个更小的子回路，这会导致其经过某个顶点两次

  以及一些判断是的充分条件：

  - **Dirac’s theorem**：如果图是连通简单图，有3个以上结点，任意结点的度数大于结点数的一半

    也即deg(v) >= n / 2

    那么该图存在汉密尔顿回路

  - **Ore’s theorem**：如果图是连通简单图，有3个以上结点，任意一对不邻接的顶点的度数和大于n，那么该图也有汉密尔顿回路

    如果都大于n-1，那么该图有汉密尔顿 通路

  Kn完全图显然存在Hamilton回路，并且是多条，任选即可

- 汉密尔顿回路用于解决点遍历问题，比如PPT中例子为跳马图，每个格子就是一个顶点，两个顶点邻接的条件是通过马的跳跃能到
- 无向完全图Kn有（n-1）/2条无公共边的汉密尔顿回路

4.6 最短路径问题

- dijkstra算法

  该算法是用于简单连通带权图的，找两个结点间最短路径

  时间复杂度为O(n^2)
- 旅行商问题（Traveling salesman problem）

  对于n个城市的TSP

  要检验 $\frac{(n-1)!}{2}$ 种汉密尔顿回路

  对于旅行商问题的同类问题，暂时没有多项式时间复杂度的解法，在阶乘的复杂度情况下，就算检验每个汉密尔顿回路只需要纳秒级的时间，也需要花上千万年去检验25个城市的旅行商问题

4.7 平面图

- 一个图任意移动边，使得任意两条边不相交，那么该图就为平面图
- 应用于电路设计
- 一个平面图可以将图分为多个区域
- 欧拉公式

  v - e + r = 2

  推论
  - 如果图G是连通的简单平面图，那么

    e ≤ 3v -6

    证明用到：

    2e ≥ 3r 带入欧拉公式即可

    但是满足e ≤ 3v -6，未必是平面图，比如K33二分图
  - 如果G是连通的简单平面图，那么必定有一个不超过五度的结点
  - K5完全图不是平面图
  - 如果G是连通的简单平面图，没有长度为3的回路，那么e ≤ 2v -4

    证明用到 2e ≥ 4r

    其实与上同理，没有3的回路，那么每个面的出入度至少为4
- Kuratowski定理

同胚：在平面图G上，如果在某条边上加多了顶点，并且将原来的一条边分别通过这个顶点变成两条边，这个操作叫元素细分，所有通过元素细分得到的图G'便是G的同胚图，如下所示

- 定理1

  如果一个图包含有与K3,3或者K5同胚的子图，那么该图不是平面图（充要条件）

- 应用，判断不是平面图

  如果确定和K5相似还是和K3,3相似，那么只用选5个或者6个顶点，6个顶点的话分成两组，可以根据度数要求，移除掉一些顶点，比如K5，移除掉度数非4的顶点，而K3,3的话移除掉非3的顶点，然后一些边是可以删除的，只要子图中含有和K3,3或者K5同胚的，那么原图就不是平面图

- 例题

  对于右侧这个图，可以看成两个三角形画出来，然后找两个三角形之间是怎样关联起来的

判断这个是否为二分图，其实很简单，一圈一圈地分组进去就好

平面图一眼丁真

欧拉图要满足都是偶度数

汉密尔顿图要满足度数大于二分之n或者两个不邻接的顶点度数和大于n

和想法一致，不确定的只是上图的回路边数，确实边数都大于3，因为经过顶点了就算1条边

此题和想法略有差异

我的想法是，由图易知，顶点数即为五边形的总数乘以5，然后面数题目已给出，再去求边数

而答案是所有面边数之和除以2（每条边都用了两次），然后已知面数再去求顶点数

4.8 图的着色问题

- 邻接区域不着色
- 平面对偶图

  其实也就是把地图用图表示，面着色换为点着色问题

- 四色定理

  一个平面图的色数不会超过4

对于环图，如果n为偶数，需要两种颜色，交替使用即可

如果n为奇数，需要三种颜色，交替使用，但最后一个点要用另一种颜色

- 目前最快的找到色数的算法需要指数级的时间
- Km,n图的色数为2

5.1 树的介绍

- 简单来说，树就是没有回路的连通无向图
- 一个图，只要没有回路，就可以称为森林，将其看作多棵树即可，连通的子图就是一棵树
- 有内点和树叶（度数为1）之分
- 定理1

  一个无向图是树的充要条件为，任意两节点有且仅有一条路径
- 根树

  指定一个树的一个结点为根节点，其它结点便都有从该结点起的路径
- n叉树，每个结点的儿子数不超过n，如果每个内点的儿子数都是n，那么称该树是满的

  上图为满三叉树
- 有序的根树

  也就是左右儿子是要和根节点比较的

这里还不太清楚，在PPT的45页左右

- 树的性质
  - n个结点的树有 n - 1条边
  - 有 i 个内点的满m叉树有 `n = m * i + 1`个结点

    利用上述公式，如果已知结点数，也可求出内点数

    另外，叶子数 `l = [(m-1) * n + 1] /m`

    由上两式可导出内点和叶子的关系

    `i = (l - 1) / (m - 1)`
- 树的高度

  从根节点出发，最长的路径长便是高度，也即深度最大的树叶对应深度

  平衡m叉树：
  - 对于高度为h的m叉树，如果每个叶子的高度都是h或者h-1，那么称其为平衡m叉树

    叶子数量 L 范围为 $2^{h-1} < L <= 2^h$

- 对于高度为h的m叉树，最多有m^h片叶子

  可以采用递归，从高度为1开始去理解，逐步拓展开，每次增加一个高度，实际上是将叶子树乘以m，所以为m的h次方

- 满树和平衡树是有差异的，满树可以不平衡，但是其内点都要有m个儿子
- 有 L 片叶子的平衡m叉树，高度大于$\log_m L$

这一块还没看完，不过不太想看了

- 应用

  利用树的结点数与边数关系以及握手定理求解

  根树的邻接矩阵的性质：

  - 对角线元素为0，易知
  - 有一列全为0的元素，这是因为有n个结点但只有n-1条边，且无向图是堆成的，那么有一列必然为0

5.2 树的应用

- 二叉搜索树
- 决策树

  找出伪造硬币的方法

5.3 树的遍历

前后中序遍历

应用于计算式的画，定义树的内点为计算符，叶子为操作数

最常用、比较清晰的是中序遍历

5.4 生成树

DFS、BFS算法以及回溯的应用

背景引入：一个连通图，包含所有顶点的最少边子图

- 生成树的概念

  图G的生成树为图G的子图，该子图为一棵树，要包含图G的所有顶点

  这里要回顾一下树的概念，其实是需要满足 e = v - 1的，并且两点之间有且仅有一条通路，图中不能有回路

  那么图中的一些边需要被去掉

- 定理1

  一个简单图是连通的 $\leftrightarrow$ 该图有生成树

  连通 -> 有生成树：不断去掉成环的边直至剩 v - 1 条边

有生成树，那么已经连通，显然加上任意边仍连通

- 上述的定理1给出了一种得到生成树的方法，这是通过减边得到的，另外，DFS同样可以得到生成树，但却是加边得到的

  通过该算法选出的边叫做树边，没有选到的叫回溯边

- BFS算法得到生成树的过程可以看作顺序，每一层往下调用，利用队列的结构操作
- 而DFS算法其实是一个逆序返回吧，从一个结点出发往下进行DFS一直到访问完所有结点

5.5 最小生成树

- 最小生成树的定义：

  一个简单连通带权图中，其生成树边权总和最小的一棵生成树

  有两种算法去寻找最小生成树

- Prime 算法

  逐渐加入与已有的顶点关联的边，要求该边连接新的结点，并且要选择多条邻接边中权值最小的，直到包含了所有顶点便完成了最小生成树查找

  时间复杂度为$O(n\log n)$

- Kruskal 算法

  逐渐加入权值最小的边，不构成环即可，直到包含所有顶点

习题课

Horner算法计算多项式

只需要n次乘法和n次加法