第一次课讲义

# 图的保存

邻接矩阵

优势：简单，符合人类正常思维习惯；可以用来进行floyed或者求可行边条数的操作

邻接表vector

【此处讲一下vector】{

.push\_back()

[]下标操作

内在机制及复杂度分析

}

优势：速度快，实现容易

劣势：在处理一些问题时不容易处理

邻接表 前向星

优势：调用时速度比vector快

劣势：建表时速度慢，且增删边开销大

-优化:链式前向星

优势：内存占用小，建表速度最快，数组模拟指针构造链表，调用速度也足够快

劣势：构造较为复杂，变量多，因为硬件原因，导致实际上调用时会比vector慢一些

其中前向星不要求掌握，矩阵和表的vector实现比较简单，所以课后只要求掌握链式前向星的模板。不能理解的话背下来也行。代码及注释已在附件中，为了写注释，所以排版比较松散。

# 最小生成树

# -给出完全图用最小的开销连接所有点

Kruskal

复杂度：mlogm

并查集+sort排序

【此处讲一下重载运算符，和sort的比较函数重写】

并查集上学期已经讲过，所以此处就跳过了。

算法本身在以上组建熟练掌握之后没有难度。

Prim

复杂度：n\*n->nlogn

【与dijkstra的异同？在讲完dijkstra之后讲该部分。】{

异

松弛操作不同：

Dijkstra：min(dis[to],dis[from]+weight)

Prim:min(dis[to],weight)

同

算法思想相同：

本质上都是贪心法，选取局部最优解，且保证局部最优解也是全局最优解。

}

要求在上课前掌握这俩算法，可用模板题LUOGU P1546测试。上课主要讲一下证明和复杂度分析。

# 单源最短路

Dijkstra【见源代码】复杂度O(n^2)

堆优化 复杂度O((n+m)logn)

-找点出队nlogn

-点进队mlogn(每次松弛都有可能成功)

核心:三角形不等式min(dis[to],dis[from]+weight)

证明 【贪心法】

SPFA

复杂度kE(平均情况k=2左右)

但是众所周知，spfa，他死了

因为对于特定的数据，复杂度最坏情况可达到O(VE)。因为论文本身证明不太严谨，我们只这里不展开讲他的复杂度。