大数据算法

第六讲 外存图数据算法

哈尔滨工业大学
王宏志
wangzh@hit.edu.cn

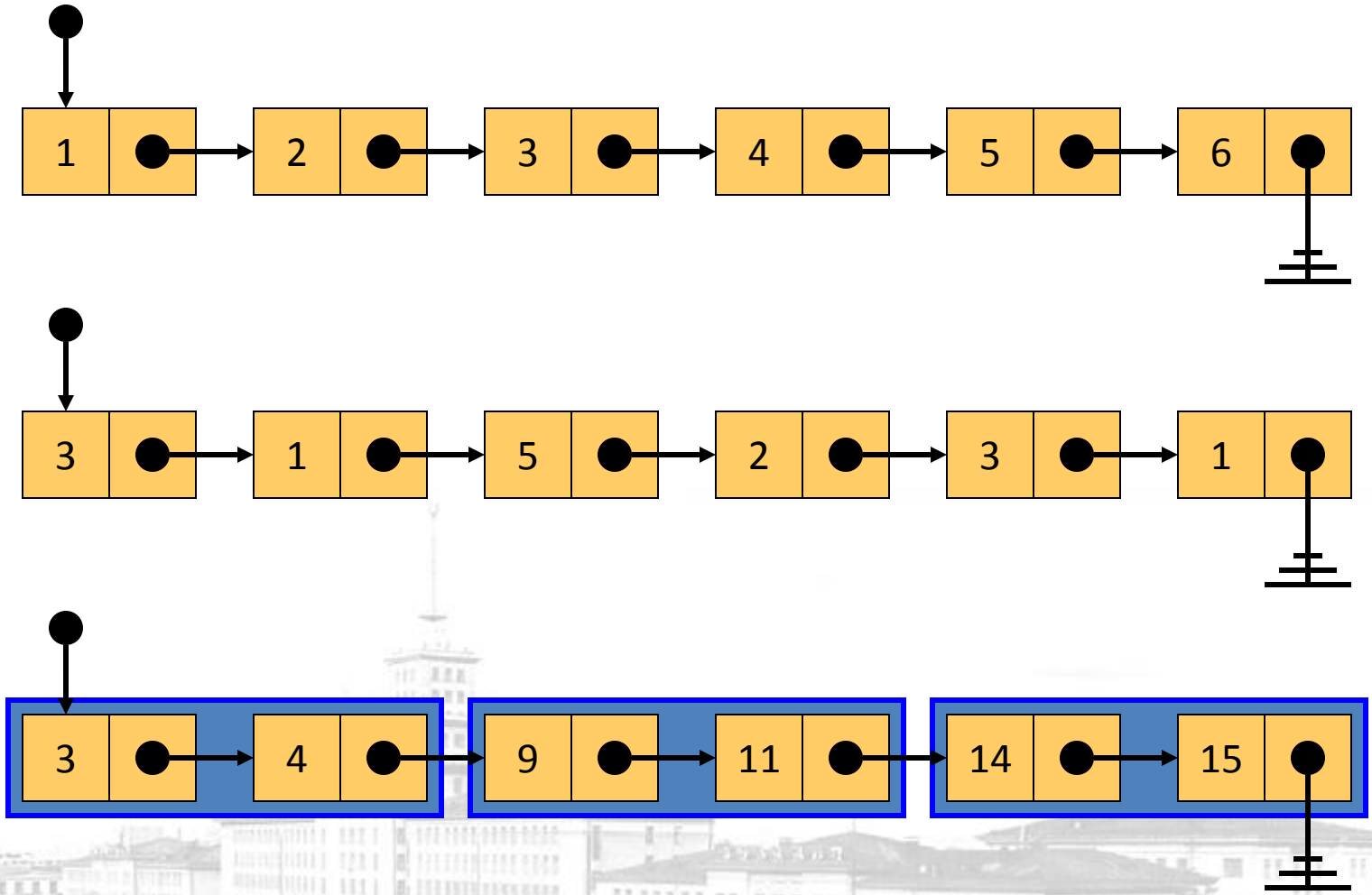
本讲内容

6.1 表排序及其应用

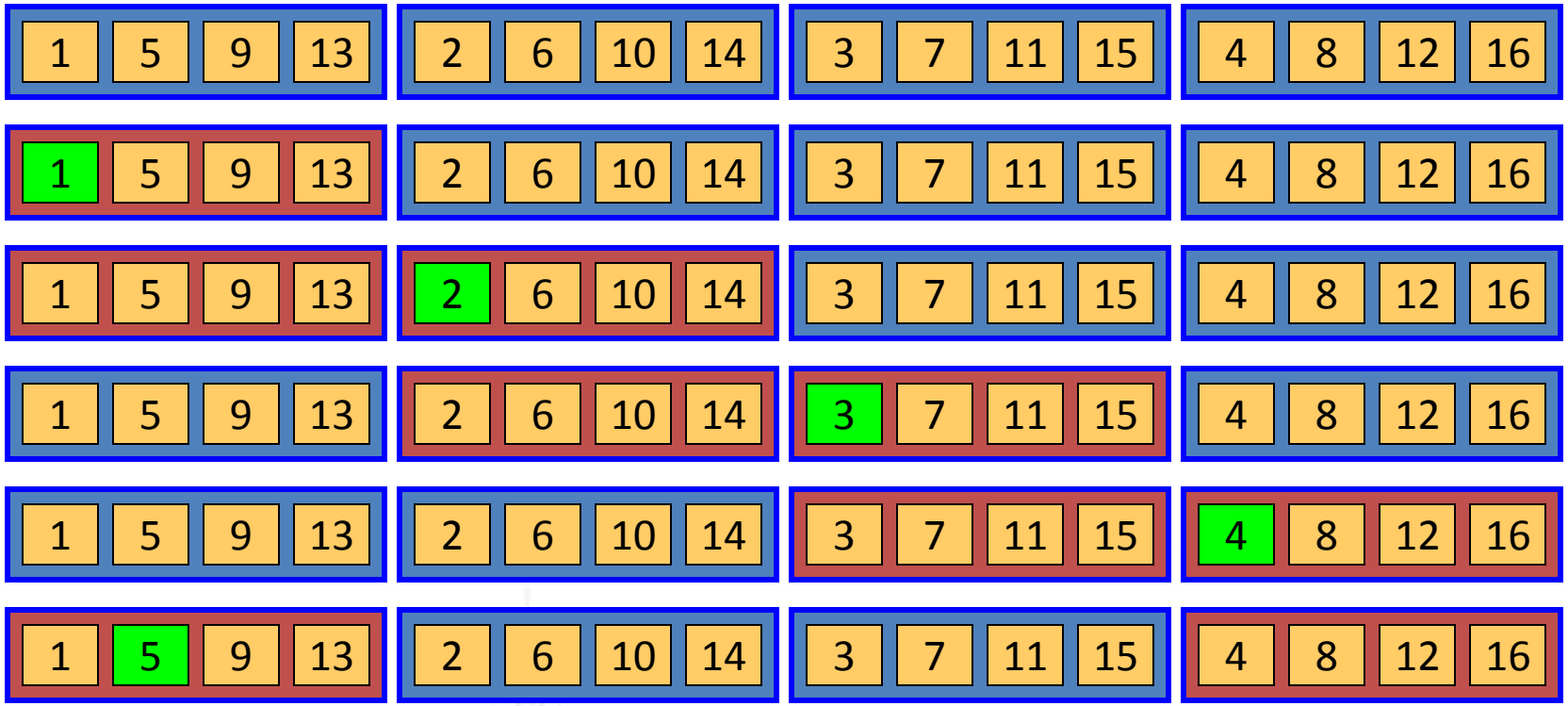
6.2 时间前向处理方法

6.3 缩图法

表排序List Ranking



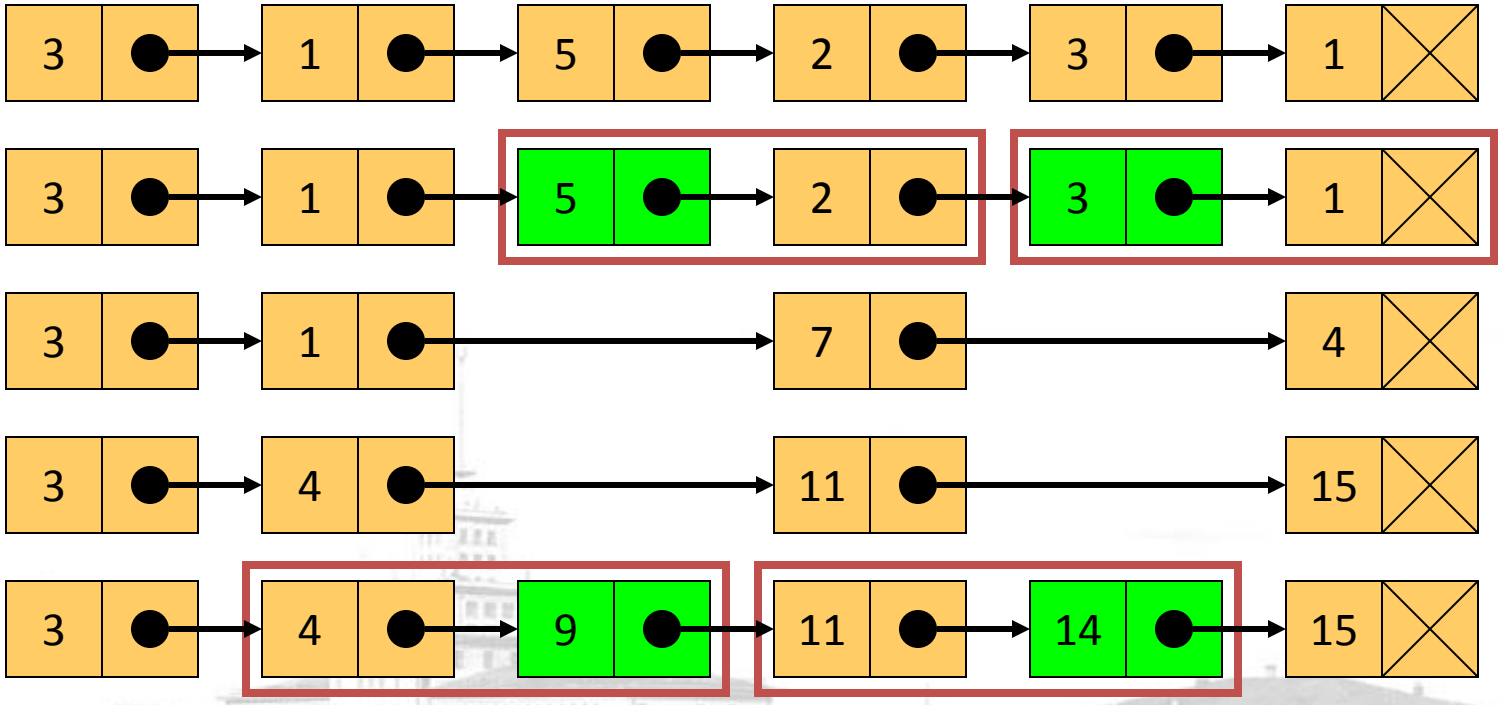
表排序的困难之处



- 内存算法最坏情况的I/O数为Ω(N)

一种高效的表排序算法

- 假设一个独立集大小至少为N/3，其中元素能高效查找(只需O(sort(N))次I/O).



一种高效的表rank算法

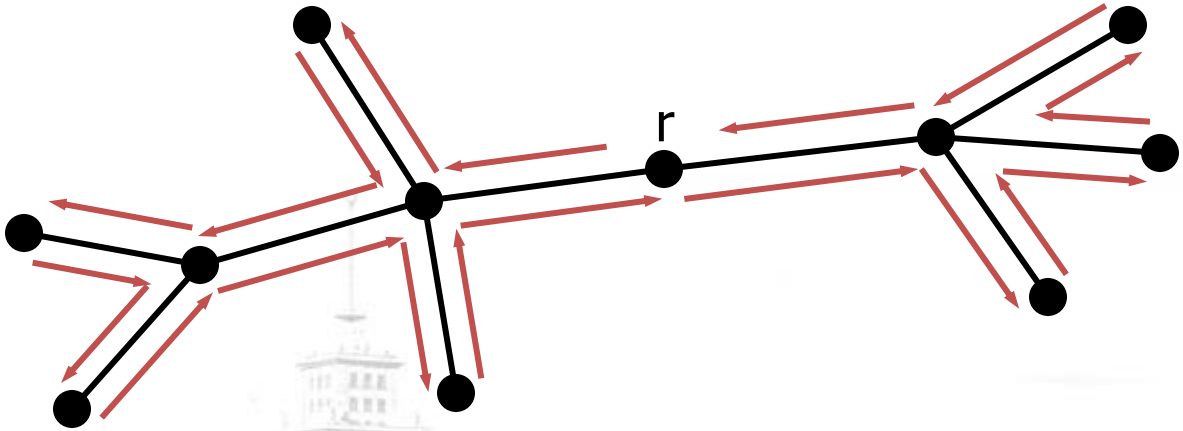
- **压缩L:**
  - 对L \ I中的元素进行排序
  - 根据后继指针对I排序
  - 对I中的每一个元素，扫描两个列表以更新succ(v)的标签
- 该过程的I/O复杂度为

$$I(N) \le I(2N/3) + O(sort(N)) = O(sort(N))$$

定理: 对大小为N的表进行rank的I/O复杂度为O(sort(N)).

欧拉回路技术

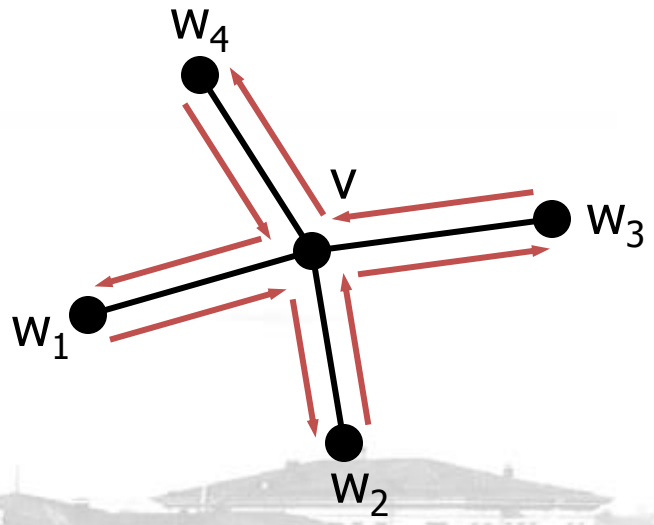
目标: 对给定的树T，以表L表示，进而让对T的每一种计算可用对L的一种rank来完成.



欧拉回路技术

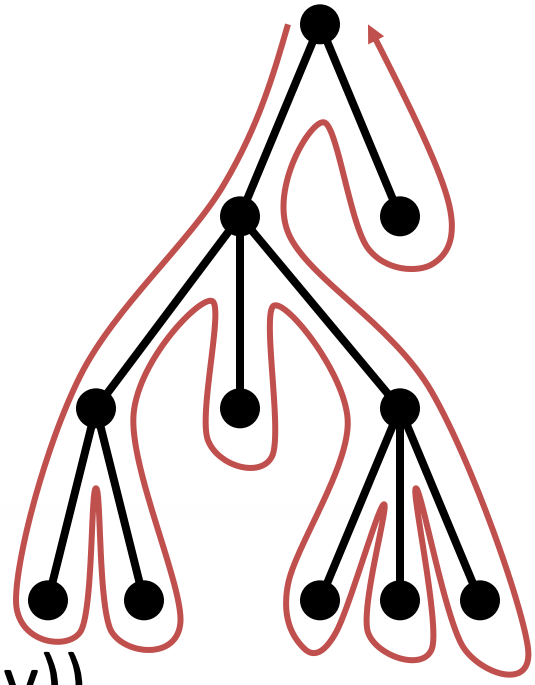
定理: 对给定的顶点邻接链表T, 其一个欧拉回路可以以O(scan(N))IO复杂性求得

- 假设$\{v,w_1\},\ldots,\{v,w_r\}$为以$v$为顶点的边
- 则$succ((w_i,v)) = (v,w_{i+1})$



父子关系判定

- 选定某结点作为根，要求确定每条边两个结点的父子关系
- 对每条边{v,w}判定亲子关系

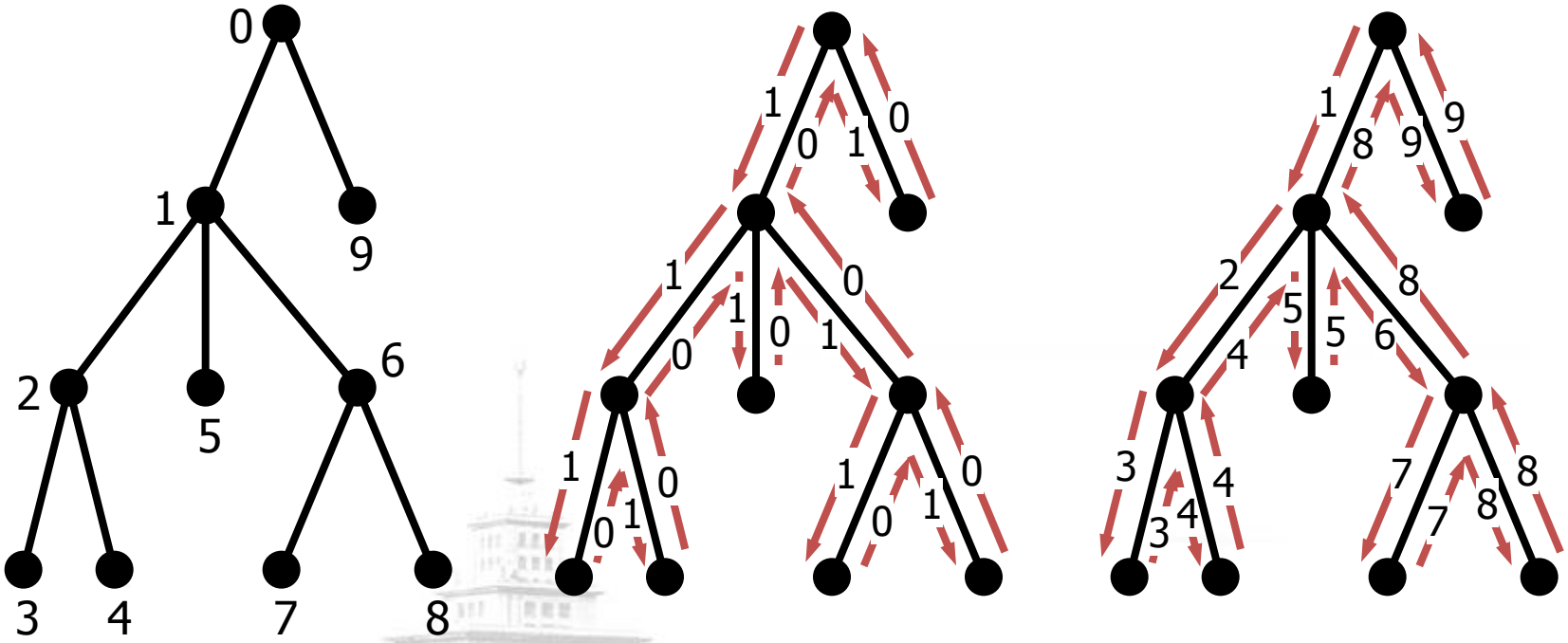


- v = p(w) 当且仅当rank((v,w)) < rank((w,v))

定理: 对树求父子关系的I/O复杂度为O(sort(N)).

计算前序计数

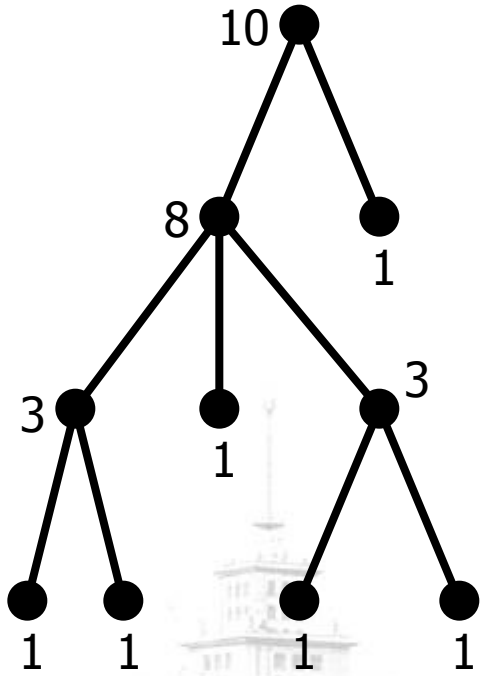
定理: 前序计数的I/O复杂度为O(sort(N)).

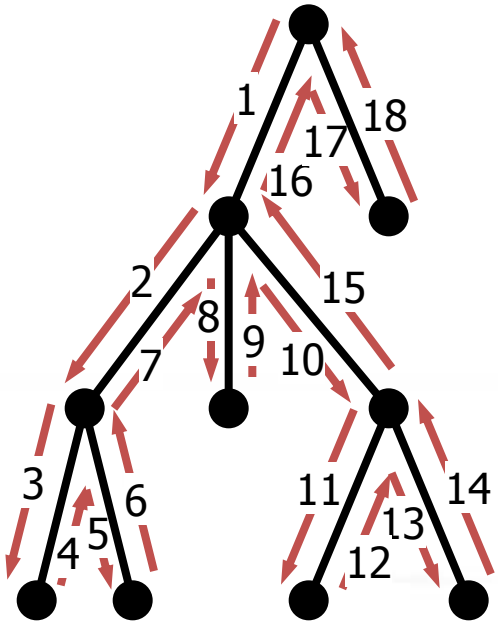


preorder#(v) = rank((p(v),v))

计算子树大小

定理: 为树T的每个结点标上子树大小的I/O复杂度为O(sort(N)).





$$|T(v)| = \frac{\text{rank}((v,p(v))) - \text{rank}((p(v),v)) + 1}{2}$$

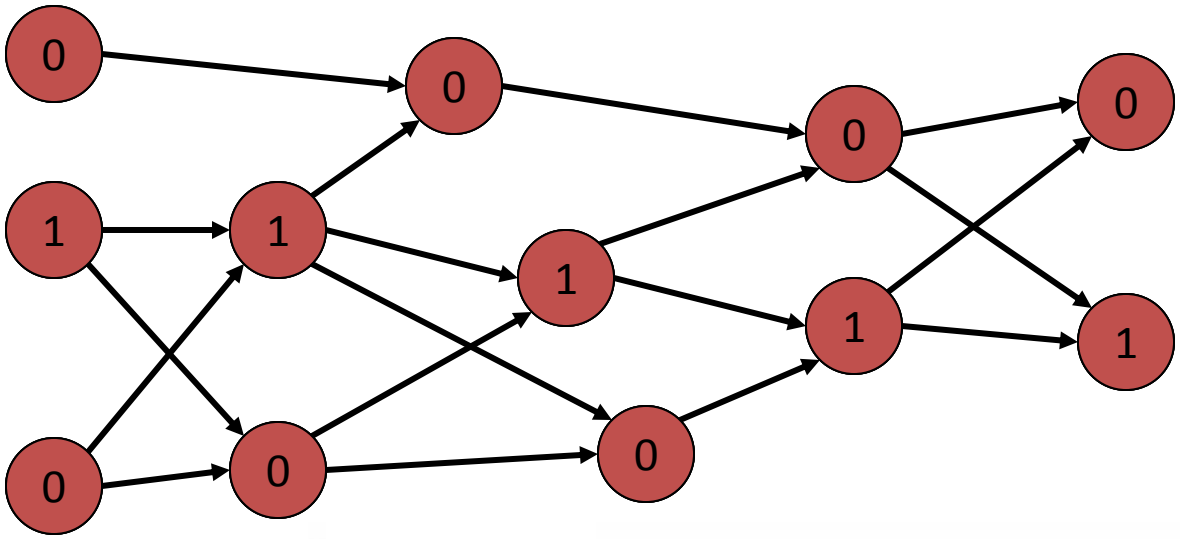
本讲内容

6.1 表排序及其应用

6.2 时间前向处理方法

6.3 缩图法

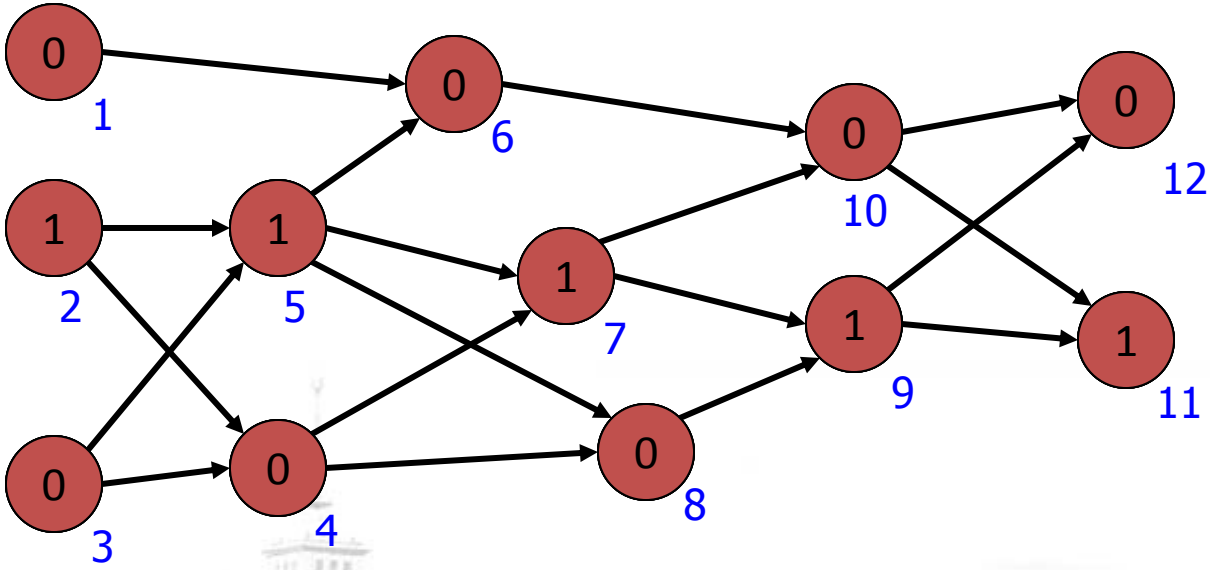
DAG上的计算



- 更一般的情况:给定标签 ϕ, 计算标签 ψ 从而 $\psi(v)$ 从$\phi(v)$ 和$\psi(u_1),\ldots,\psi(u_r)$中计算出来, 其中 $u_1,\ldots,u_r$ 是v的入邻居

时间前向的处理

- 假设结点已经按照拓扑排序给出
  - ➢ 利用优先队列Q沿着边传送数据



Q:

时间前向的处理

分析:

- 扫描对象：点集合 + 邻接链表

➢I/O复杂度O(scan(|V| + |E|))

- 优先队列:
  - ➢每条边从队列中进出一次
  - ➢O(|E|)个优先队列操作

➢I/O复杂度为O(sort(|E|))

定理: 一个DAG G = (V,E) 可以用O(sort(|V| + |E|))次I/O计算.

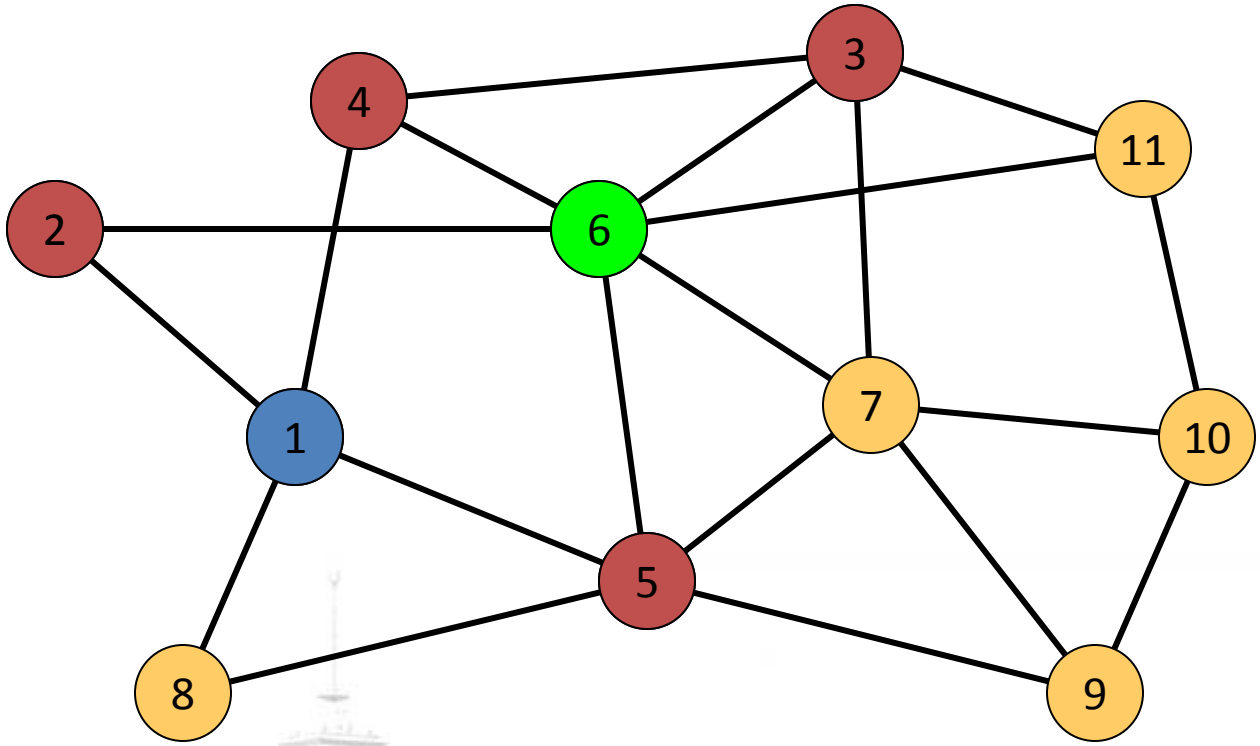
最大独立集 (MIS)

伪代码**Algorithm** GREEDYMIS:

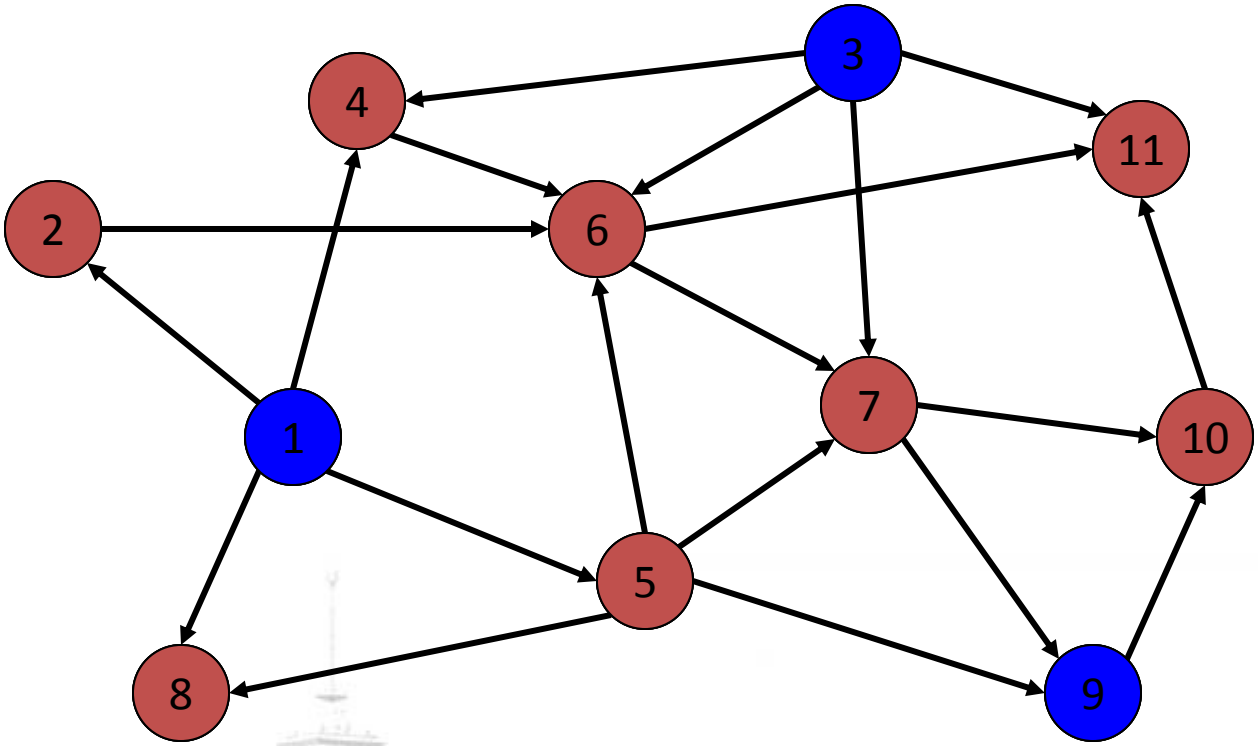
1. I ← 0
2. **for** 每个顶点v ∈ G **do**
3. **if** 在I中没有v 的邻居在 **then**
4. 将v加入I
5. **end if**
6. **end for**

观察: 考虑v的被上一轮迭代中访问的邻居就足够了.

最大独立集Maximal Independent Set (MIS)



最大独立集

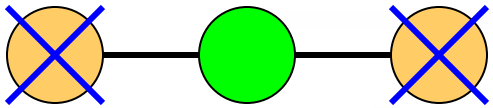


定理: 求图G = (V,E)的最大独立集的I/O复杂度为O(sort(|V|+|E|)).

表的大独立集

推论: 查找规模为N的表L中的大小至少为N/3的独立集的I/O复杂度为O(sort(N)).

- 独立集(MIS)I中的每个结点都使得其余两个结点不在I中



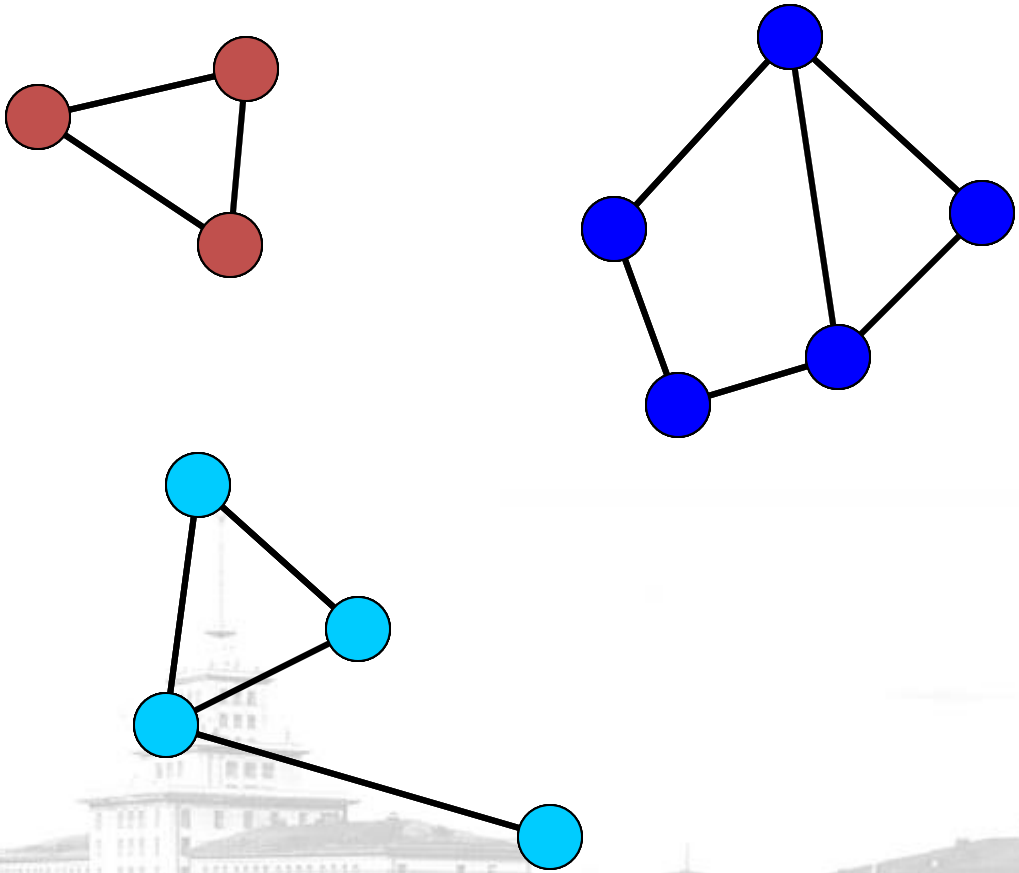
➢ 每个独立集(MIS)的大小至少为N/3.

本讲内容

6.1 表排序及其应用
6.2 时间前向处理方法
6.3 缩图法

连通性

半外存算法



连通性

半外存算法

分析:

- 扫描顶点集，以将顶点载入主存
- 扫描边集以执行算法
- I/O复杂度为O(scan(|V| + |E|))

定理: 假设$|V| \leq M$，求图的连通分量的I/O复杂度为O(scan(|V| + |E|))

连通性

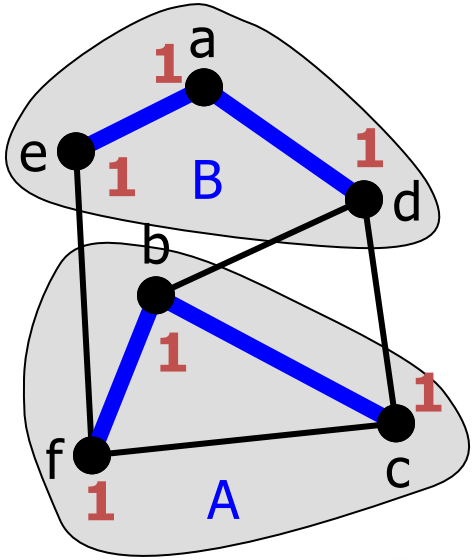
一般情况

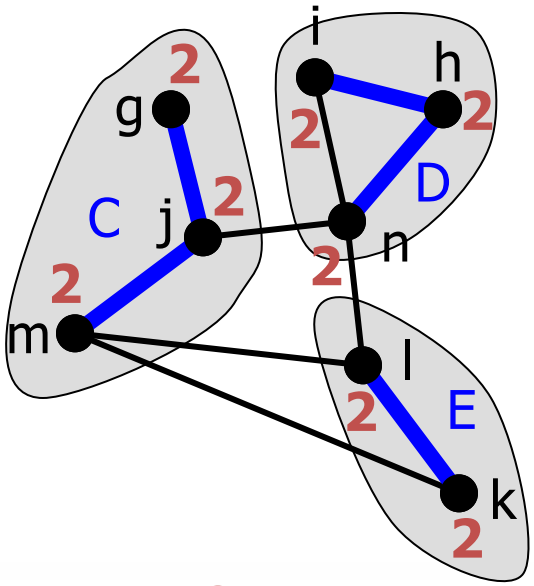
思想:

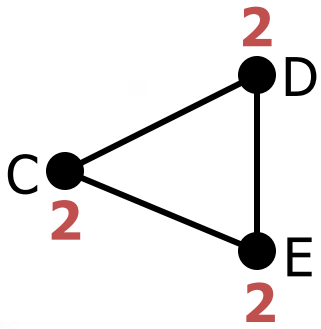
- $|V| \leq M$
  - 可使用半外存算法(semi-external algorithm)
- $|V| > M$
  - 找到G的简单连通子图
  - 收缩上述子图得到图$G' = (V', E')$ 其中 $|V'| \leq c|V|$, $c < 1$
  - 递归计算G'的连通分量
  - 从G'中得到G的各个连通分量

连通性Connectivity

一般情况







连通性Connectivity

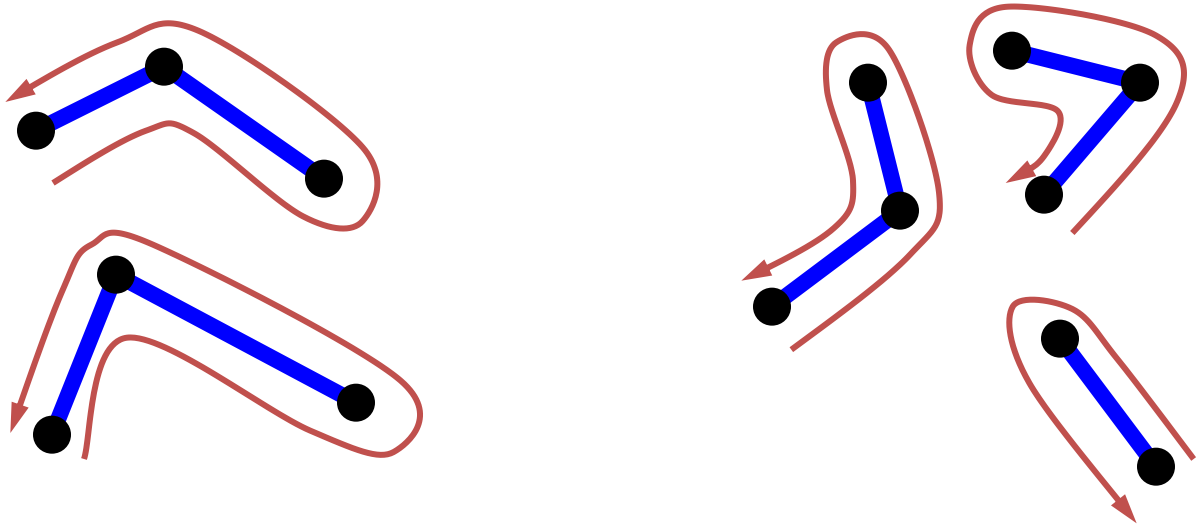
一般情况

主要步骤:

- 对于每个点找到最小的邻居(容易)
- **计算图H由选定的边导出的连通分量**
- 将每个连通分量缩为一个结点(容易)
- 递归调用上述过程
- 对每个$v \in G'$，将其连通度复制到其代表的G中的每个结点上(容易)

连通性

一般情况



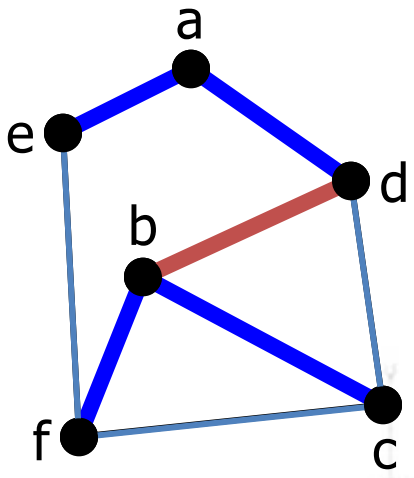
- 每个强连通分量H的大小至少为2
  - $|V'| \le |V|/2$
  - $\log(|V|/M)$ 次递归调用

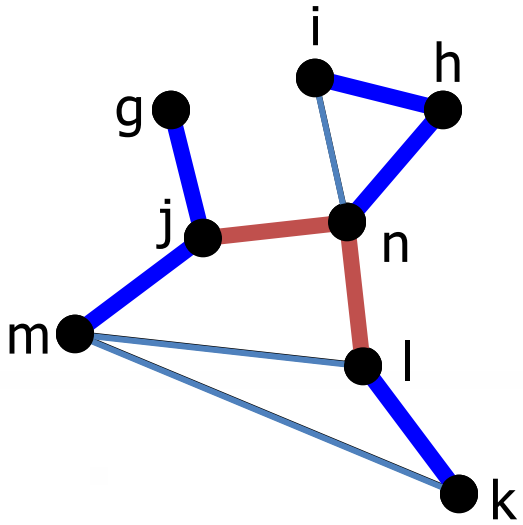
定理: 计算图 G = (V,E) 的连通分量的I/O复杂度为

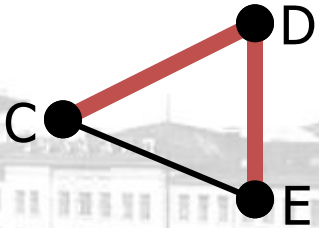
$$O(sort(|E|)\log(|V|/M))$$

最小生成树 (MST)

观察: 图的连通性算法可扩增为求图G最小生成树的算法



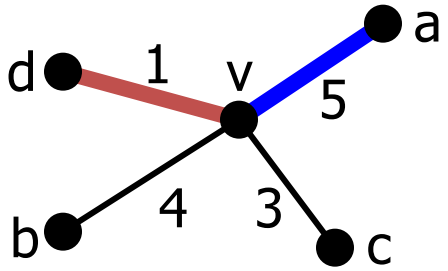




最小生成树 (MST)

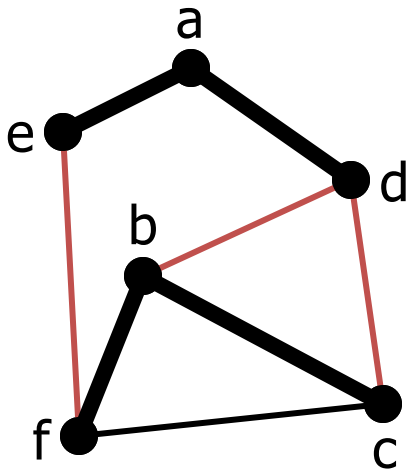
获得最小生成树的过程:

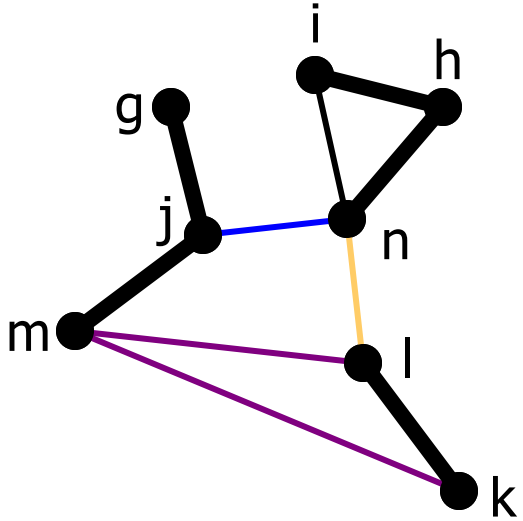
- 选择权值最小且与v相关的边

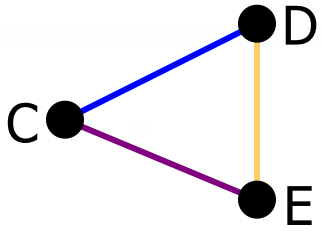


- 一些注解:
  - 压缩后的图中 某条边的权值= 该边代表的所有边的权值的最小值
  - 当树加入边e时，实际上加入了这条最小边

最小生成树 (MST)







定理: 计算图G = (V,E)的最小生成树的I/O复杂度为

$$O\left(sort(|E|)\log\left(\frac{|V|}{M}\right)\right)$$

图算法的三种技术

- **时间前向处理:**
  - 将图问题表示为有向无环图的估值问题
- **缩图法:**
  - 保持其他特定的情况下，缩减图G 的规模
  - 在压缩图中递归解决问题
  - 根据压缩图的解，构造图 G的解
- **自举:**
  - 一旦输入(或其部分)规模足够小，即可切换为相对低效的算法

致谢

- 本讲义部分内容来自于Norbert Zeh的讲义