4.4.2 最小生成树

无向连通带权图G=(V,E,W)，$w(e)\in W$是边e的权. G的一棵生成树是包含了G的所有顶点的树, 树中各边的权之和称为树的权，具有最小权的生成树称为G的**最小生成树**.

命题4.1 设 G 是 n 阶连通图，那么

(1) T 是 G 的生成树当且仅当 T 有 $n-1$ 条边.

(2) 如果 T 是 G 的生成树，$e\notin T$，那么$T\cup\{e\}$含有一个圈(回路).

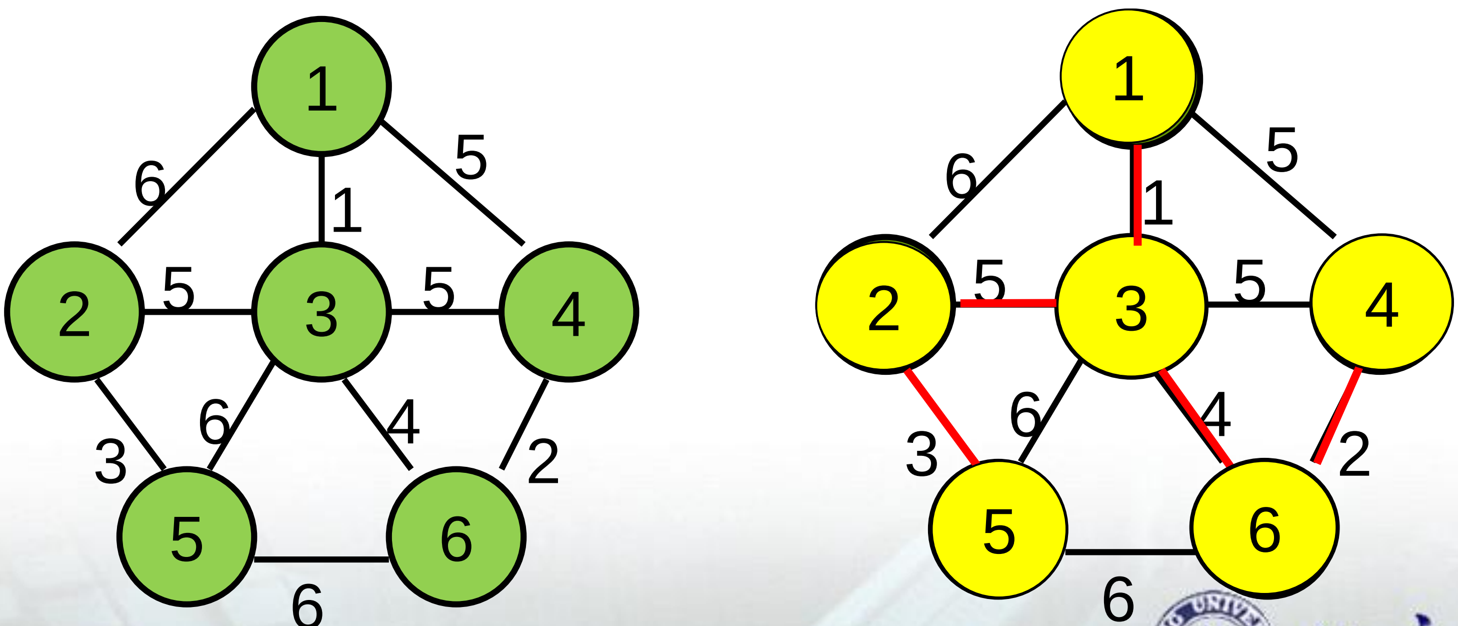
问题：给定连通带权图G，求G的一棵最小生成树.

算法：**Prim算法**和**Kruskal算法**

Prim算法

算法 Prim(G,E,W)

1. $S \leftarrow \{1\}$
2. **while** $V-S \neq \varnothing$ **do**
3. 从$V-S$中选择j使得j到S中顶点的边权最小
4. $S \leftarrow S \cup \{j\}$



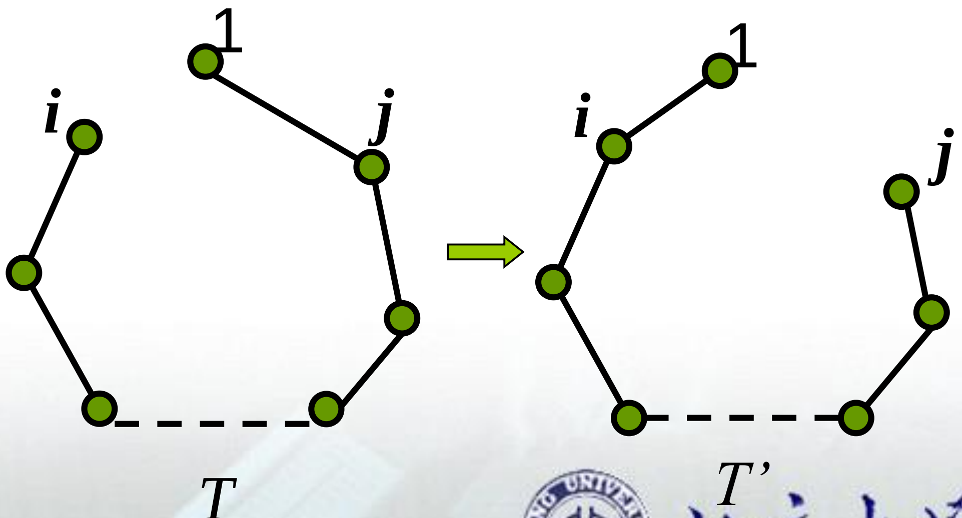
正确性证明

对步数归纳

定理：对于任意 $k < n$，存在一棵最小生成树包含算法前 k 步选择的边

归纳基础：$k=1$，存在一棵最小生成树 T 包含边 $e=(1,i)$，其中 $(1,i)$ 是所有关联 1 的边中权最小的.

设 T 为一棵最小生成树，假设 T 不包含 $(1,i)$，则 $T\cup\{(1,i)\}$ 含有一条回路，回路中关联1的另一条边为 $(1,j)$，令 $T' = (T-\{(1,j)\})\cup\{(1,i)\}$，则 T' 也是生成树，且 $W(T') \le W(T)$.



正确性证明(续)

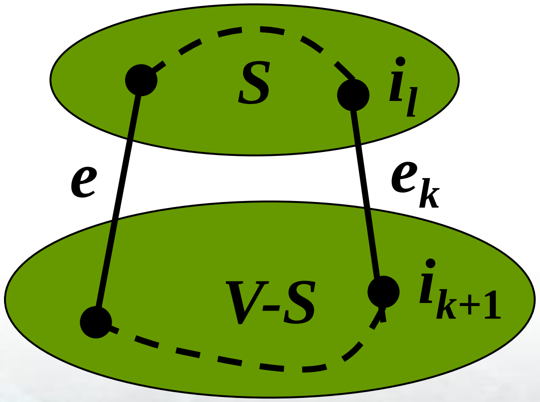
归纳步骤：

假设算法进行了$k-1$步，生成树的边为$e_1,e_2,\ldots,e_{k-1}$，这些边的 k 个端点构成集合S. 由归纳假设存在G 的一棵最小生成树T 包含这些边.

算法第k 步选择了顶点 i_{k+1}, 则 i_{k+1}到S中顶点的边权最小, 设这条边为 $e_k=(i_{k+1},i_l)$. 假设T不含有e_k，则将 e_k 加到T 中形成一条回路. 这条回路有另外一条连接S与$V-S$中顶点的边e, 令

$$T^*=(T-\{e\})\cup\{e_k\},$$

则T^*是G 的一棵生成树，包含$e_1,e_2,\ldots,e_k$, $W(T^*)\le W(T)$.



算法时间：$T(n)=O(n^2)$

Kruskal算法

算法4.6 Kruskal

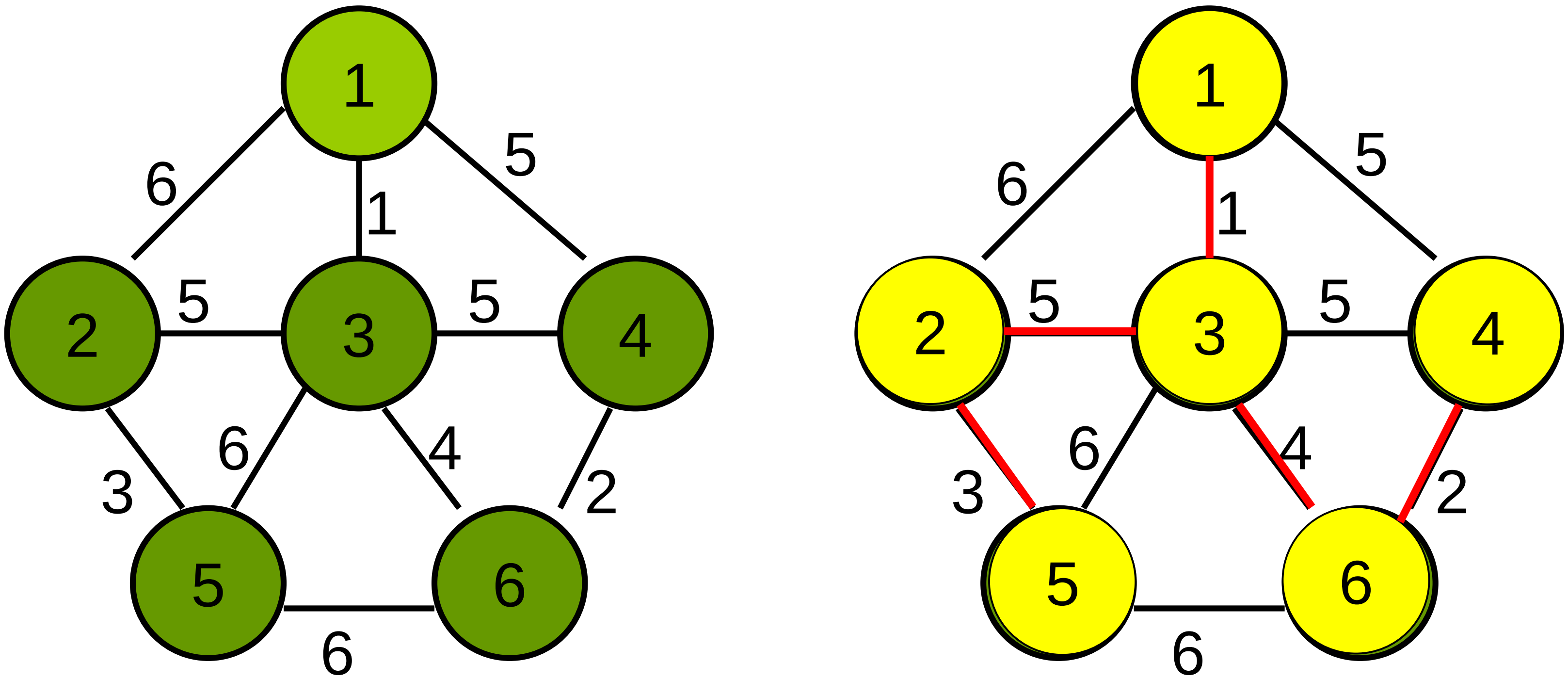
输入：连通图*G*　　　　// 顶点数*n*，边数*m*

输出：*G*的最小生成树

1. 按权从小到大排序G中的边，使得$E=\{e_1, e_2, \ldots, e_m\}$
2. $T \leftarrow \varnothing$
3. repeat
4. 　　$e \leftarrow E$中的最短边
5. 　　if e的两端点不在同一个连通分支
6. 　　then $T \leftarrow T \cup \{e\}$
7. 　　$E \leftarrow E - \{e\}$
8. until T包含了$n-1$条边

北京大学

实例



Kruskal算法正确性证明

命题：对于任意 $n>1$, 算法对 n 阶图得到一棵最小生成树.

证明 $n=2$, 只有一条边，命题显然为真.
假设对于n个顶点的图算法正确，考虑 $n+1$个顶点的图G, G中最小权边 $e=(i,j)$，从G 中短接 i 和 j，得到图G'. 根据归纳假设，由算法存在G' 的最小生成树T'.令$T=T'\cup\{e\}$, 则T 是关于G 的最小生成树.

否则存在G 的含边e 的最小生成树T^*，$W(T^*)<W(T)$. (如果 $e\notin T^*$, 在T^*中加边e，形成回路. 去掉回路中任意别的边所得生成树的权仍旧最小). 在T^*中短接 e 得到G' 的生成树$T^*-\{e\}$, 且

$$W(T^*-\{e\})=W(T^*)-w(e)<W(T)-w(e)=W(T'),$$

与T' 的最优性矛盾.

算法的实现与时间复杂度

数据结构：

建立FIND数组，FIND[i] 是结点 i 的连通分支标记.

(1) 初始FIND[i]=i.

(2) 两个连通分支合并，则将较小分支结点的FIND值更新为较大分支的标记

时间复杂度：

(1) 每个结点至多更新log$n$次，建立和更新FIND数组的总时间为$O(n\log n)$

(2) 算法时间为

$$O(m\log m) + O(n\log n) + O(m) = O(m\log n)$$

边排序　　FIND数组　其他

4.4.3 单源最短路径

给定带权有向网络G=(V,E,W),每条边e=<i,j>的权w(e)为非负实数，表示从i到j的距离. 源点$s \in V$，求从s出发到达其它结点的最短路径.

Dijkstra算法：

$x \in S \Leftrightarrow x \in V$ 且从 s 到 x 的最短路径长度已知

初始：$S=\{s\}$，$S=V$时算法结束

从 s 到 u 相对于S 的最短路径：从 s 到 u且仅经过S 中顶点的最短路径

$dist[u]$：从 s 到 u 的相对于S 的最短路径的长度

$short[u]$：从 s 到 u 的最短路径的长

$dist[u] \ge short[u]$

Dijkstra算法

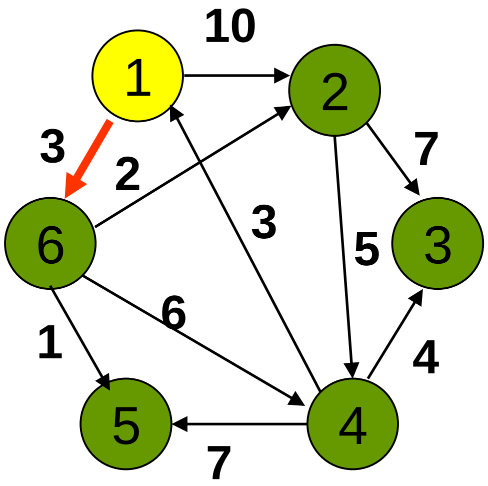
算法 **Dijkstra**

$S \leftarrow \{s\}$
$dist[s] \leftarrow 0$
for $i \in V-\{s\}$ **do**
    $dist[i] \leftarrow w(s,i)$ // 如果 s 到 i 没有边, $w(s,i)=\infty$
while $V-S \neq \varnothing$ **do**
    从$V-S$中取出具有相对S的最短路径的顶点 j
    $S \leftarrow S \cup \{j\}$;
    for $i \in V-S$ **do**
        if $dist[j]+w(j,i)<dist[i]$
        then $dist[i] \leftarrow dist[j]+w(j,i)$ // 更新$dist[i]$

实例

输入：$G=<V,E,W>$， 源点 1

$V=\{1, 2, 3, 4, 5, 6\}$

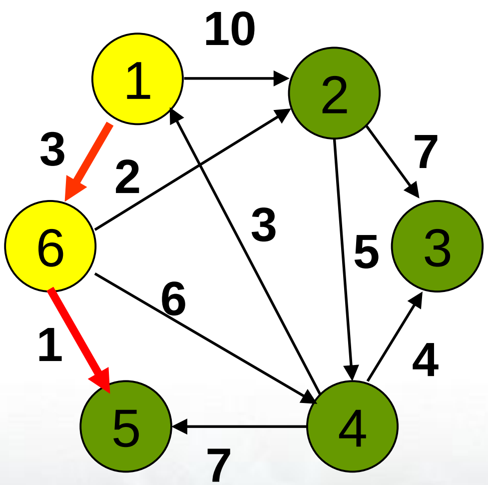


$S=\{1\}$,

$dist[1]=0$

$dist[2]=10$, $dist[6]=3$

$dist[3]=dist[4]=dist[5]=\infty$



$S=\{1,6\}$,

$dist[1]=0$, $dist[6]=3$

$dist[2]=5$, $dist[4]=9$, $dist[5]=4$

$dist[3]=\infty$

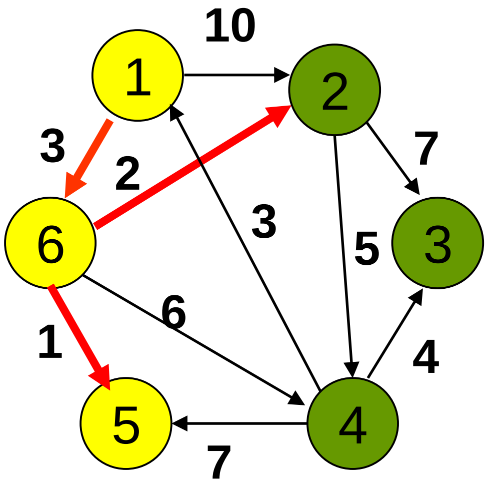
实例（续）

S={1,6,5},

dist[1]=0, *dist*[6]=3, *dist*[5]=4

dist[2]=5, *dist*[4]=9,

dist[3]=∞



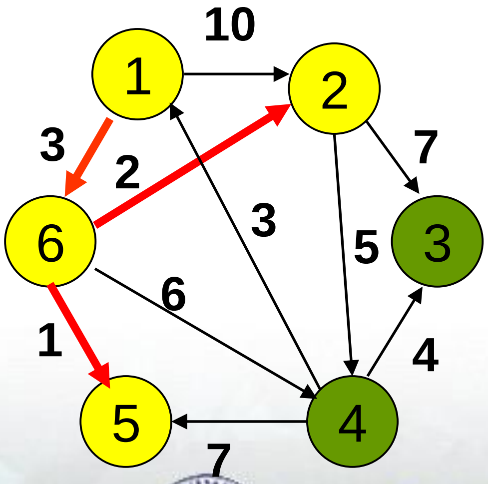
S={1,6,5,2},

dist[1]=0, *dist*[6]=3, *dist*[5]=4

dist[2]=5

dist[3]=12

dist[4]=9



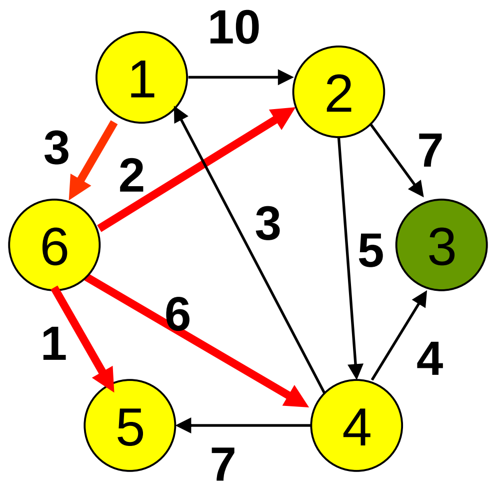
实 例（续）

S={1,6,5,2,4},

dist[1]=0, *dist*[6]=3, *dist*[5]=4

dist[2]=5, *dist*[4]=9,

dist[3]=12

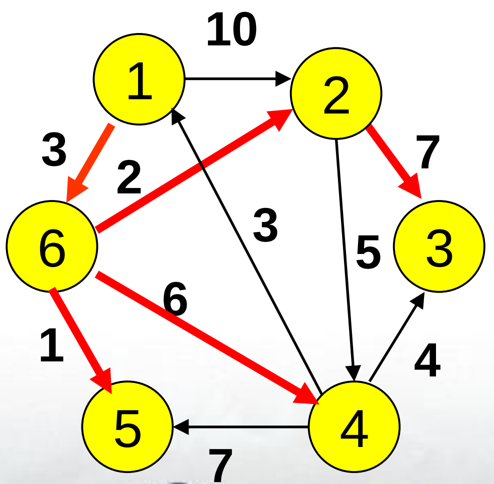


S={1,6,5,2,4,3},

dist[1]=0, *dist*[6]=3, *dist*[5]=4

dist[2]=5, *dist*[4]=9

dist[3]=12



解：

short[1]=0, short[2]=5,

short[3]=12, short[4]=9,

short[5]=4, short[6]=3.

算法正确性证明

命题：当算法进行到第 k 步时，对于S 中每个结点 i，

$$dist[i] = short[i]$$

归纳基础　k=1, S={s}, $dist[s]=short[s]=0$, 命题为真.

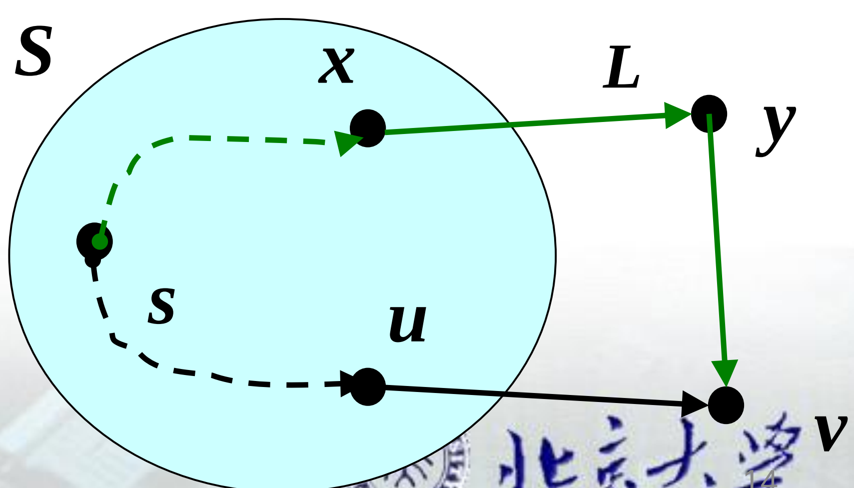
归纳步骤　假设命题对于k 为真. 考虑 k+1步, 选择顶点v (边(u,v)).　假若存在另一条 s-v 路径 L (绿色)，最后一次出S 的顶点为 x, 在这次从S 中出来后经过V–S 的第一个顶点为 y.

$dist[v] \le dist[y]$　　//v先被选

$\le dist[y]+d(y,v) \le L$

$dist[v]=short[v]$

时间复杂度 $T(n)=O(n^2)$



贪心法小结

(1) 适用于组合优化问题. 求解过程是多步判 断. 判断的依据是局部最优策略，使目标值达到最大(或最小)，与前面的子问题计算结果无关.

(2) 局部最优策略的选择是算法正确性的关键.

(3) 正确性证明方法：数学归纳法、交换论证. 使用数学归纳法主要通过对算法步数或者问题规模进行归纳. 如果要证明贪心策略是错误的，只需举出反例.

(4) 自顶向下求解，通过选择将问题归约为小的子问题.

(5) 如果贪心法得不到最优解，可以对问题的输入进行分析或者估计算法的近似比.

(6) 如果对原始数据排序之后，贪心法往往是一轮处理，时间复杂度和空间复杂度低.