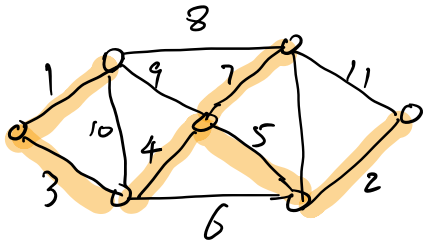
LN6. 最小生成树 (Minimum Spanning Tree, MST)

* 定义:

已知: 图 $G=(V,E)$, 无向、简单、连通图.

$\forall e \in E,\ C_e > 0$. 带权图.

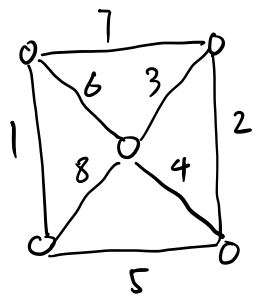


求: 连通子图 $V(T)=V$

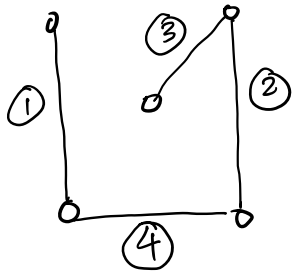
目标: 最小化 $C(T)=\sum_{e\in E(T)} C_e$.

* 性质: 无环、连通 $\Rightarrow$ 树.

* 算法:



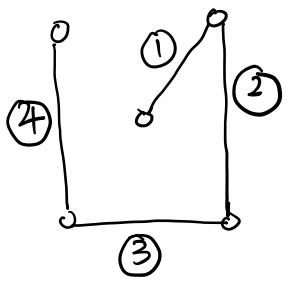
Kruskal:



保持无环.

[kill circle]

Prim:



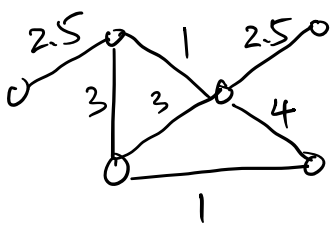
保持连通.

不断增长树.

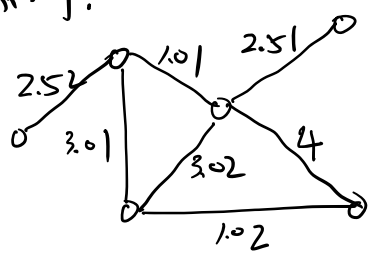
* 等长边正确性证明:

假设: 不失一般性, 假设边权均不同.

例:

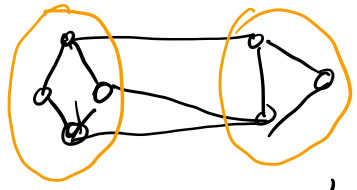


$\Rightarrow$



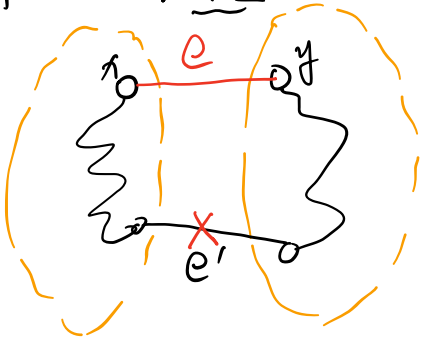
Def. 割边 (cut): S 中, S 与 V/S 之间的边构成的集合.

例:



Lemma. MST的割边性质:
对于任意的一个割 $(S, V/S)$, 其割边等中最小权的边 e
一定被包含在MST中.

Pf. 反证: 若T是一个MST. 但不包含e. 则一定存在另一条割边 e'
$c_e < c_{e'}$



加入e, 去掉 e'. 得到新的 ST, T'

$c_{(T-e)} + c_e - c_{e'} \Rightarrow c_{T'}$

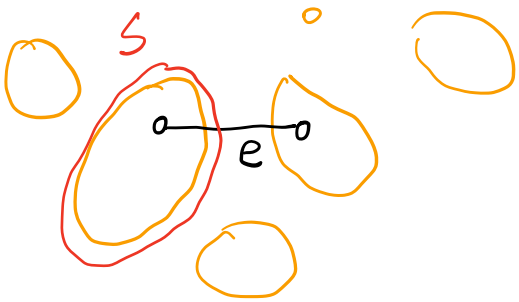
$c_{T'} < c_T$ 矛盾.

∴ 最小割边e一定在MST中.

Thm. Kruskal算法输出的是 MST.

Pf: ① ST

② 每次加入的边均满足以上lemma.
即一定是割边等中的最小权的边.

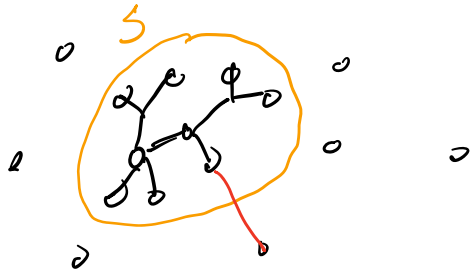


割T以为V/S

Thm. Prim算法输出的是MST.

Pf. ① ST

② 加入的每条边均满足以上Lemma.



其余点构成的集合为V/S.

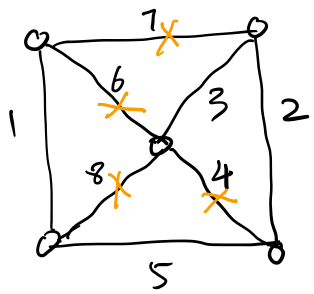
LN7. 其它的MST算法.

Alg3. 减边算法：从G出发.

每一步，在保持连通性的前提下，

删去权最大的边.

Q：如何证明. (hw)



Alg4. Blue / Red 算法.

给环
- ① Blue，加边：找一个 cut(X, $\bar{X}$)，$X \longleftrightarrow$ 一个无蓝边
  选此Cut中最大权的边加入，染为蓝色.
- ② Red，减边：找一个 cycle C，上面无红边.
  此边环中最大权的边，染为红色.

直到所有边被染色.

输出蓝边等为MST.

LN8. 降低MST算法的复杂度 ⇒ 基于并查集的MST.

* Kruskal算法的复杂度分析: $|V|=n, |E|=m$

$T=(V,\phi)$

循环. while T 不连通. $O(n)$, $n-1$次.

{ 选择连接2个子树的最小权的边 e. $O(m)$

$T=T\cup\{e\}$. $O(1)$

end

返回 T.

⇒ 复杂度 $O(mn)$

* 对 Kruskal 算法. 如何降低复杂度? 排序.

对边从小到大排序 $O(m\log n)$

循环: 对排序的边 $e=(u,v)$ m次

{ if u,v 属于不同的连通子图. ← Find(u), Find(v) $\log n$

$T=T\cup\{e\}$

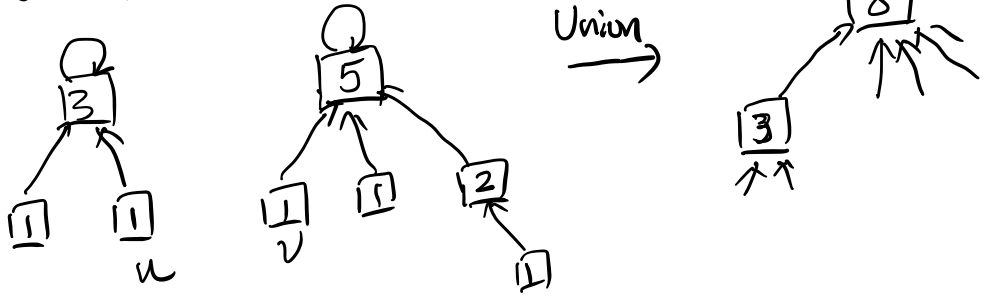
更新T的连通子图. ← Union(T(u), T(v)) $\log n$

end if

end for

输出 T.

$m\log n$

* 

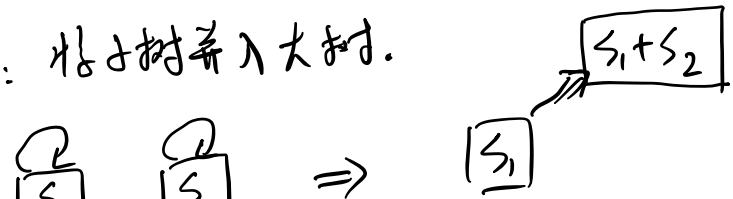
初始化：n个节点，每个点 i

Find(u)：从点u出发，找到树根，输出根节点 $O(\log n)$

Union(u,v)：$r_1 \leftarrow Find(u)$ $r_2 \leftarrow Find(v)$ $O(\log n)$

把小树并入大树，并更新根节点处树的大小。

关键：将小树并入大树。

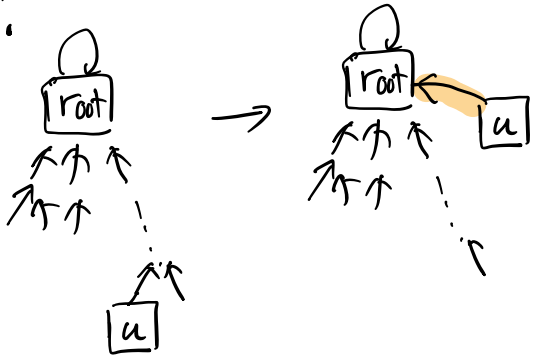


$S_1 \le S_2$. 合并树高 $\le 2 \cdot S_2$ 的高度。

$\Rightarrow$ 算法复杂度为 $O(m \log n)$.

☆ 计算 MST 的更快的算法？

路径压缩：Find(u)



补充：逆 Ackermann 函数：$A(n)$ 的逆函数 $\alpha(n)$

正向的操作：

① 后继：$x \leftarrow x+1$

② 加法：$x+y = x+\underbrace{1+1+\cdots+1}_{y\text{次}}$

③ 乘法：$x \cdot y = \underbrace{x + x + \cdots + x}_{y\text{次}}$

④ 指数：$x^y = \underbrace{x \cdot x \cdot \cdots \cdot x}_{y\text{次}}$

⑤ $x \uparrow y$：$x \uparrow y = x^{x^{x^{\cdots}}}$ 个 y 次.

$A(m,n)$：

m \ n	0	1	2	3	…	n
0	1	2	3	4	…	n+1
1	2	3	4	5	…	n+2
2	3	5	7	9	…	2(n+3)-2
3	5	13	29	61	…	$2^{n+3}-3$
4	13	65533	$2^{65533}-3$		…	$2\uparrow(n+3)-3$
5	65533	$A(4,65533)$				$2^{2^{\cdots^2}}$

对角线 $A(n)$ 第逆1的 $A^{-1}(n) = \alpha(n)$

⇒ 路径压缩的 MST：$O(m \cdot \alpha(m))$，前提：排好序

✱ 进一步探讨的问题：

Q：线性时间复杂度的 MST 算法？ Open.

Q：若边权均为整数，创等基运算时间为 $O(1)$. ⇒ $O(m)$

Q：随机算法：Kargar, Klein, Tarjan. 1995. $O(n)$.