ch8 图的优化问题和贪心算法

8.1 Introduction

- 最优化问题
  - 问题给定某些约束条件，满足这些约束条件的解称为问题的可行解。
  - 通常可行解不唯一
  - 给定目标函数，使目标函数最大（或最小）的可行解为最优解
  - 求解最优解----穷举法、贪心算法、动态规划等

贪心算法的基本思想

- 贪心算法总是做出当前看来最好的选择，并不从整体最优考虑。所作的选择只是在某种意义上的局部最优选择。
- 当然我们也希望贪心算法得到的结果在整体上也是最优的。
- 虽然贪心算法不是对所有问题都能得到整体最优解，但是对一些问题他还是能得到最优解；对一些问题即使得不到最优解，也能得到近似最优解。

8.1 Introduction

- 贪心算法
  - 特点：是一步一步地进行，常以当前情况为基础根据某个优化测度作最优选择，而不考虑各种可能的整体情况，它省去了为找最优解要穷尽所有可能而必须耗费的大量时间
  - 它采用自顶向下，以迭代的方法做出相继的贪心选择，每做一次贪心选择就将所求问题简化为一个规模更小的子问题，通过每一步贪心选择，可得到问题的一个最优解
  - 虽然每一步上都要保证能获得局部最优解，但由此产生的全局解有时不一定是最优的，所以贪婪法不要回溯。

贪心算法的基本要素

- 最优子结构性质：一个问题的最优解包含其子问题的最优解，那么此问题具有最优子结构性质。
- 贪心选择性质：问题的整体最优解可以通过一些列局部最优的选择（贪心选择）来达到。因此贪心算法和分治法一样，都是自顶向下的思想。

8.1 Introduction--贪心选择性质

指所求问题的整体最优解可以通过一系列局部最优的选择，即贪心选择来达到。这是贪心算法可行的第一个基本要素，也是贪心算法与动态规划算法的主要区别。

动态规划算法通常以自底向上的方式解各子问题，而贪心算法则通常以自顶向下的方式进行，以迭代的方式作出相继的贪心选择，每作一次贪心选择就将所求问题简化为规模更小的子问题。

对于一个具体问题，要确定它是否具有贪心选择性质，必须证明每一步所作的贪心选择最终导致问题的整体最优解。

8.1 Introduction--最优子结构性质

当一个问题的最优解包含其子问题的最优解时，称此问题具有**最优子结构性质**。问题的最优子结构性质是该问题可用动态规划算法或贪心算法求解的关键特征。

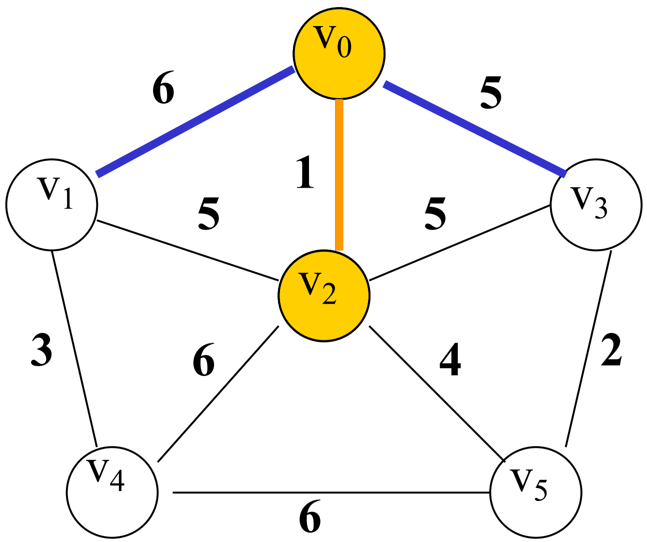
Prim算法的基本步骤

- $G=(V, E)$是连通网，TE是G上最小生成树中边的集合。
- 初态：$U=\{u_0 | u_0 \in V\}$, $TE=\{\}$ 开始，重复执行下述操作：

  1）在所有$u \in U$，$v \in V-U$的边$(u,v) \in E$中找一条带权最小的边(u_0,v_0)并入集合TE, v_0并入U。

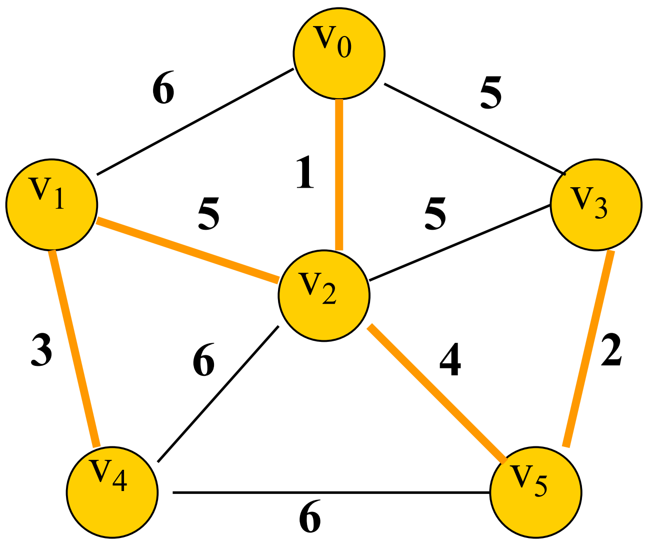
  2）重复1）直至U=V为止。此时TE中必有n-1条边，则T=（V，TE）为G的最小生成树。

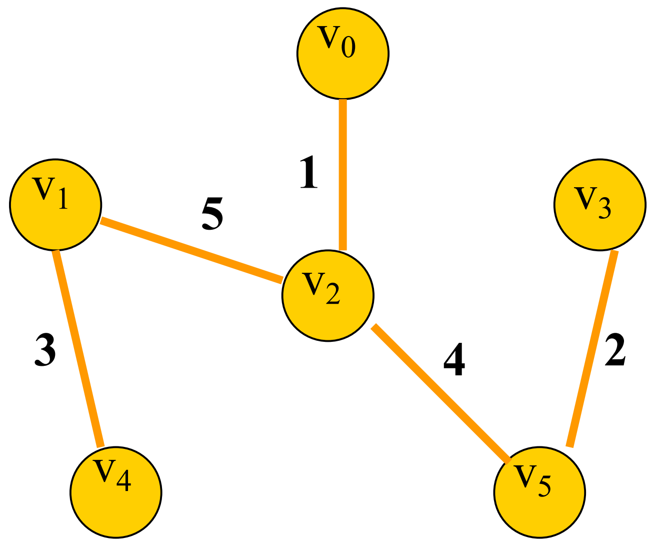
8.2 prim—最小生成树

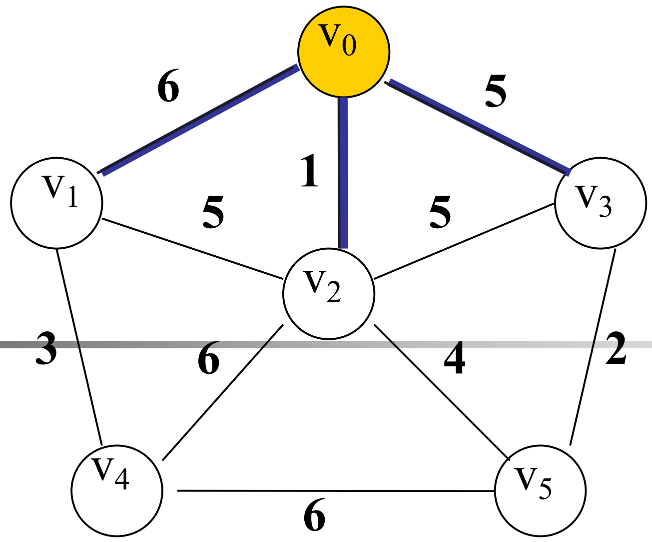


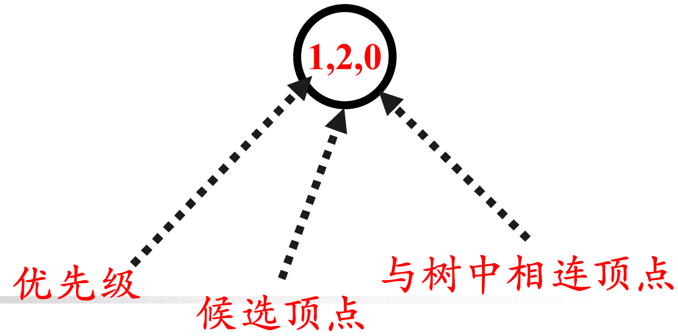
1. *tree* vertices: in the tree constructed so far,
2. *fringe* vertices: not in the tree, but adjacent to some vertex in the tree,
3. *unseen* vertices: all others.

8.2 prim——最小生成树







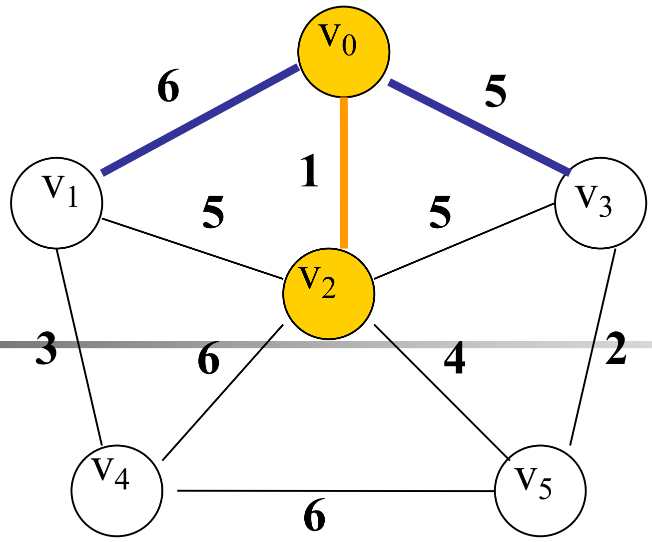


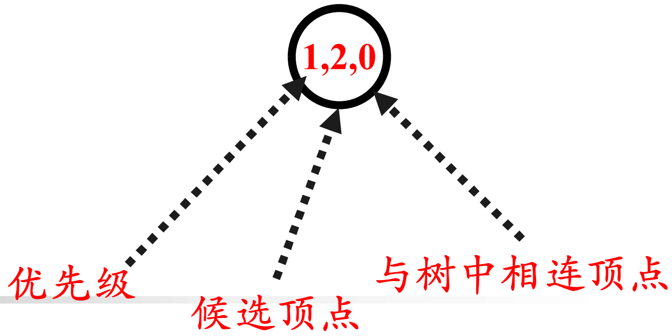
insert

insert

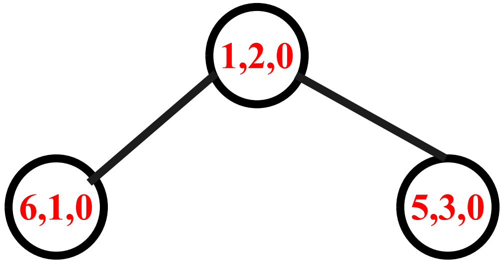
insert

最小优先队列

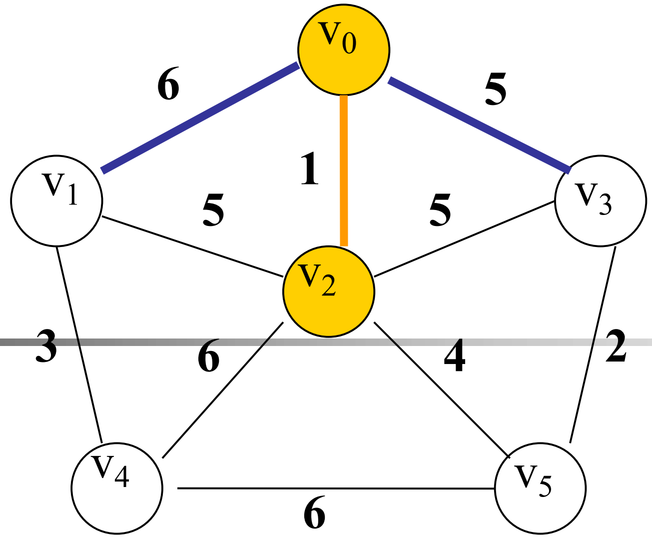


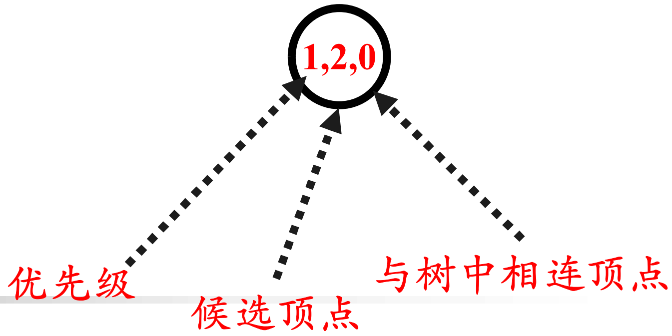


getMin取优先级最小的顶点及其相连的边加入生成树



最小优先队列

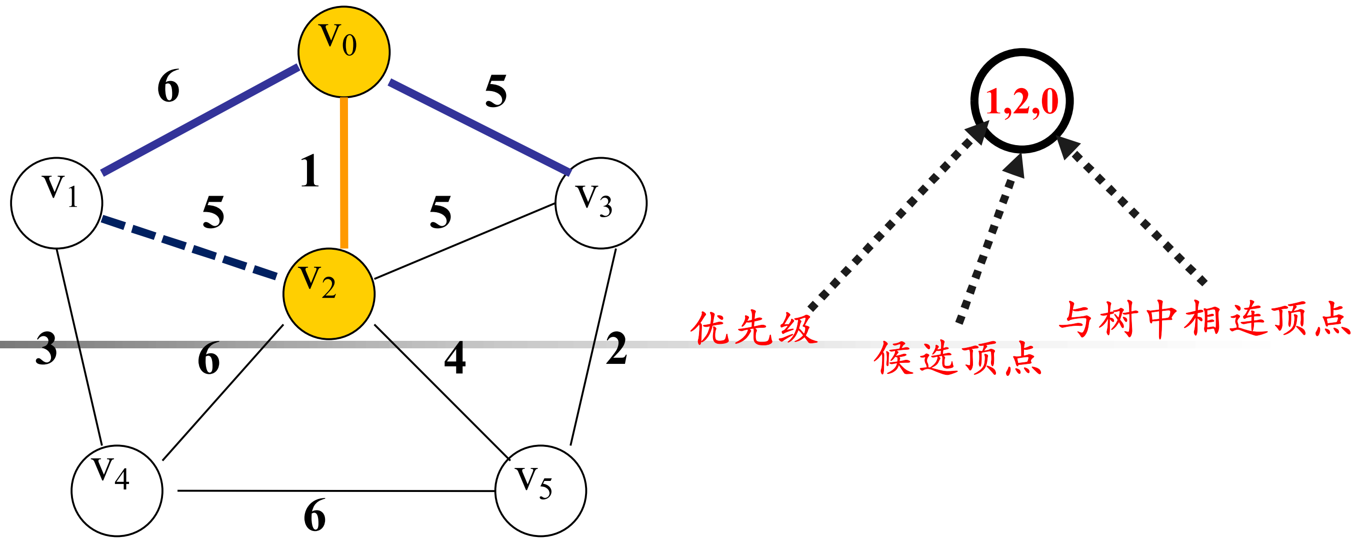




getMin取优先级最小的顶点及其相连的边加入生成树

deleteMin

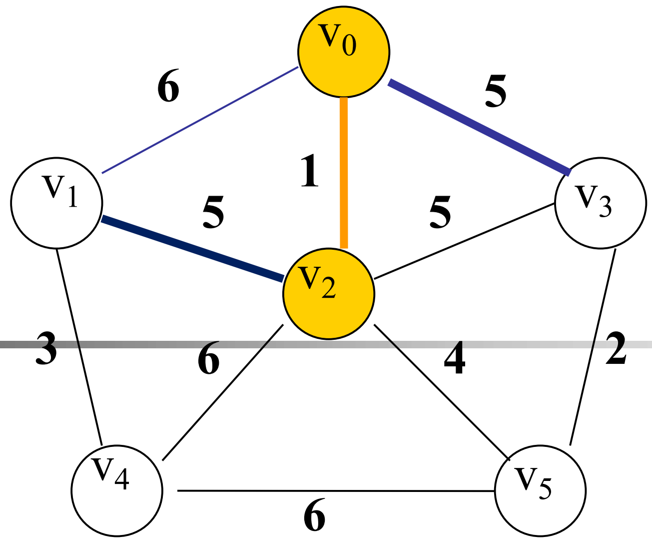
最小优先队列

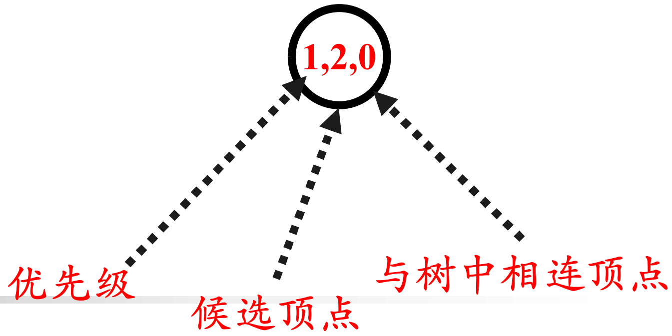


DeacreaseKey

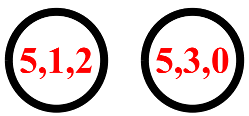
5,1,2 5,3,0

最小优先队列

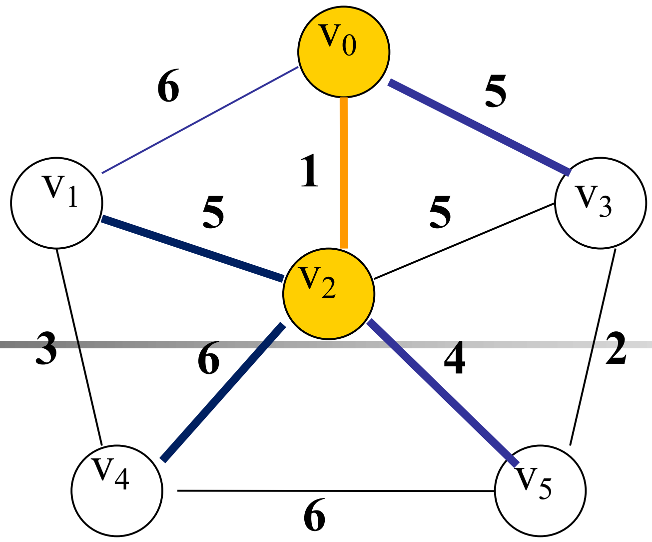


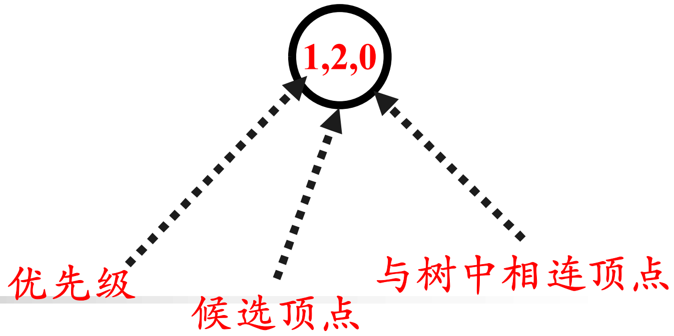


DeacreaseKey



最小优先队列



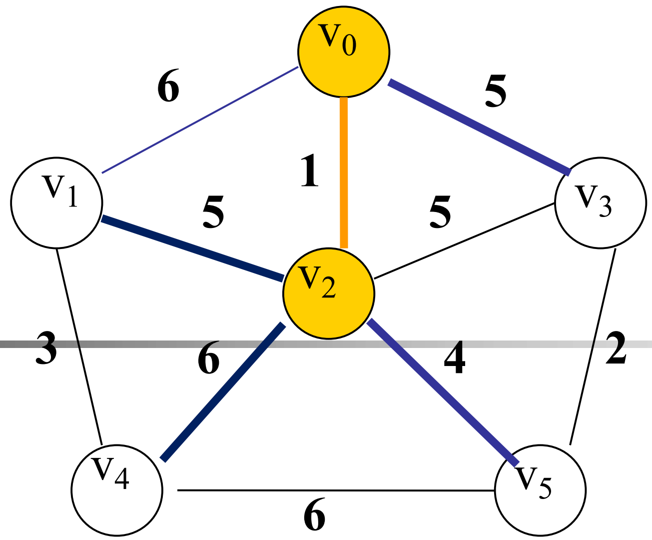


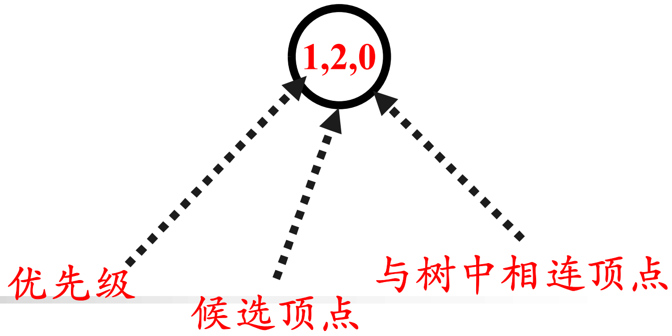
insert

insert

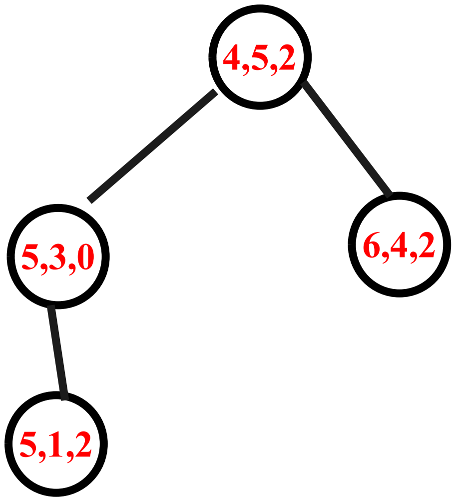
4,5,2 6,4,2 5,1,2 5,3,0

最小优先队列

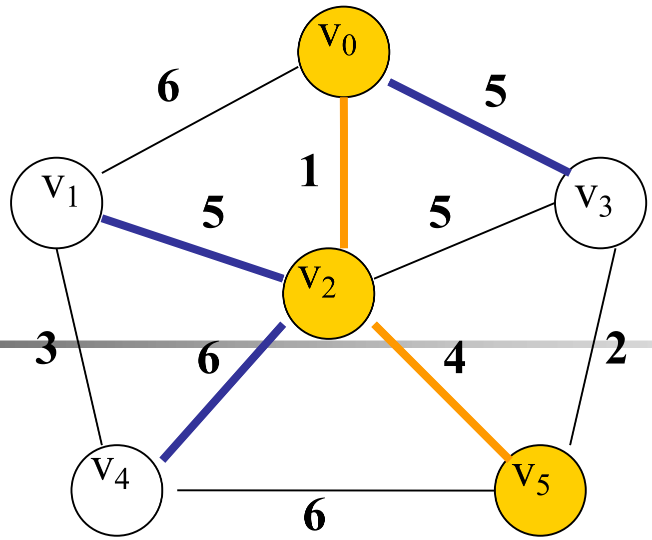


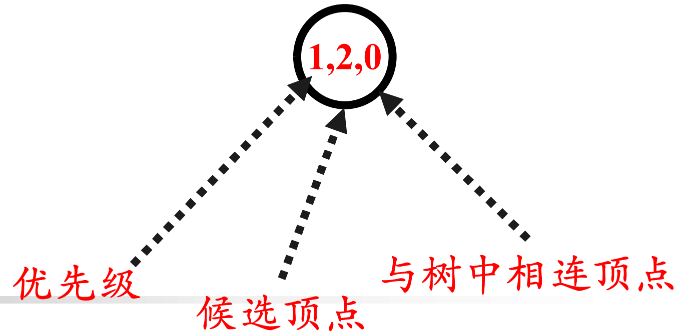


getMin取优先级最小的顶点及其相连的边加入生成树



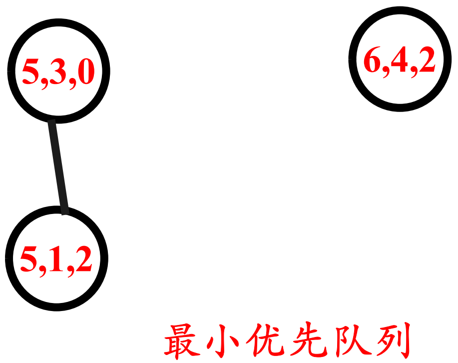
最小优先队列

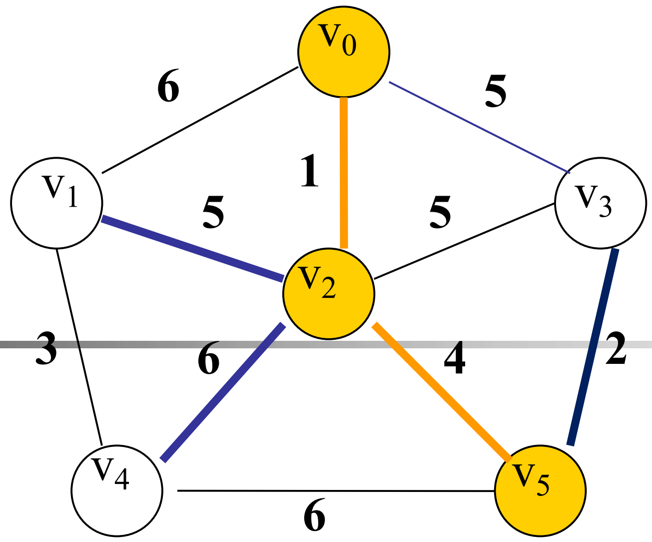


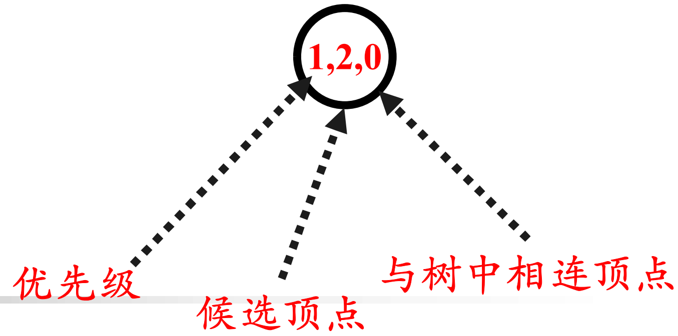


getMin取优先级最小的顶点及其相连的边加入生成树

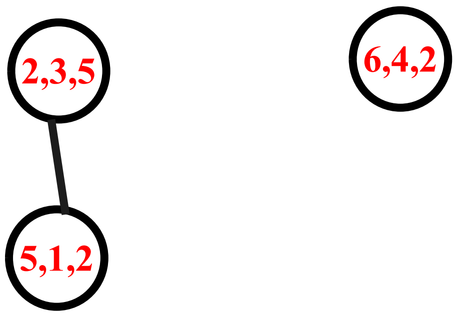
deleteMin



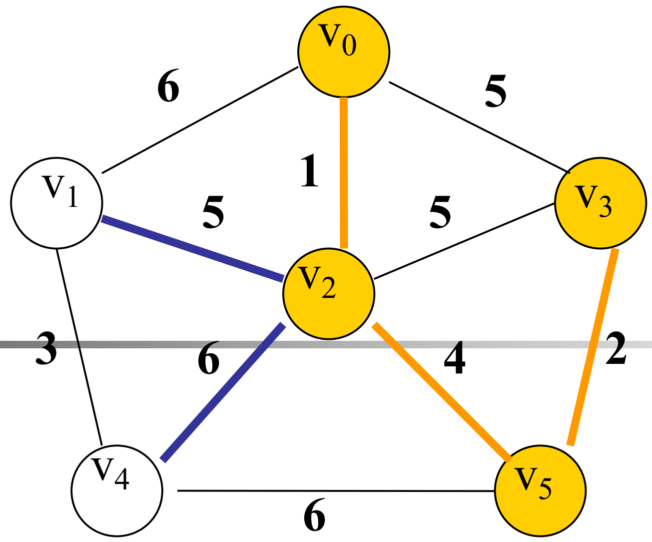


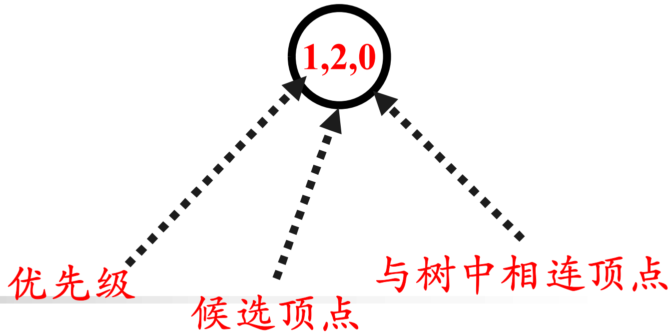


decreseKey



最小优先队列

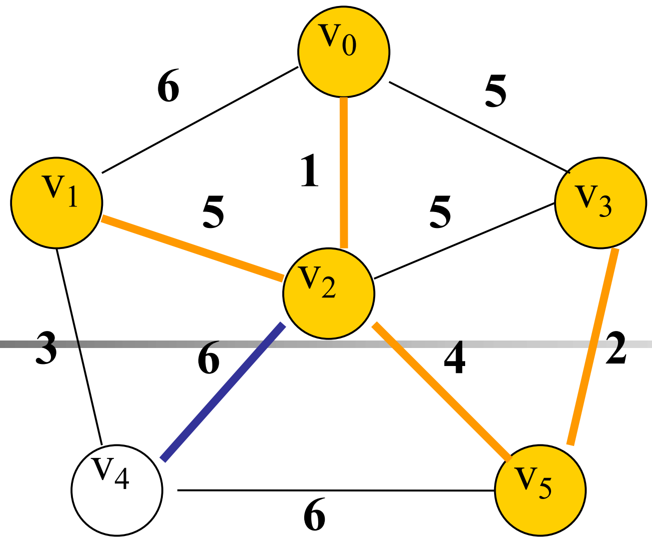


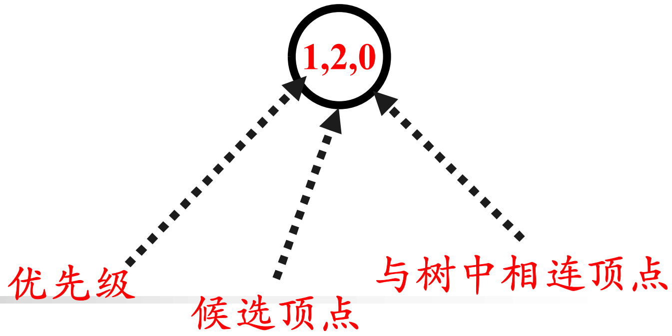


getMin取优先级最小的顶点及其相连的边加入生成树

deleteMin

最小优先队列

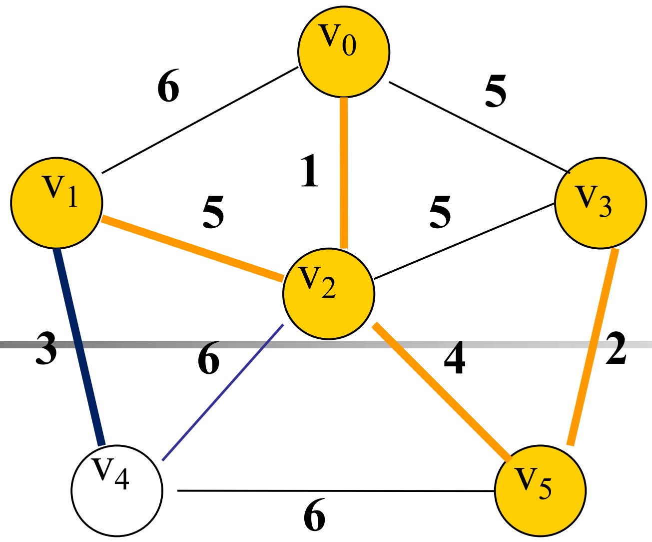


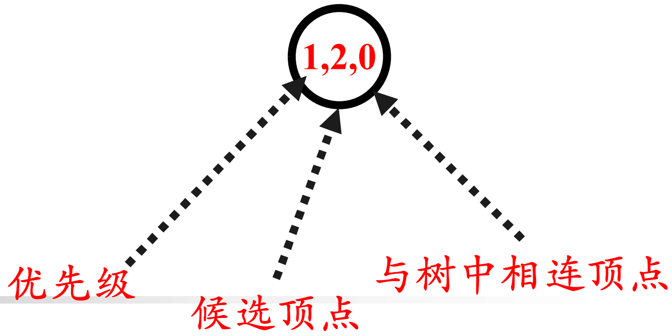


getMin取优先级最小的顶点及其相连的边加入生成树

deleteMin

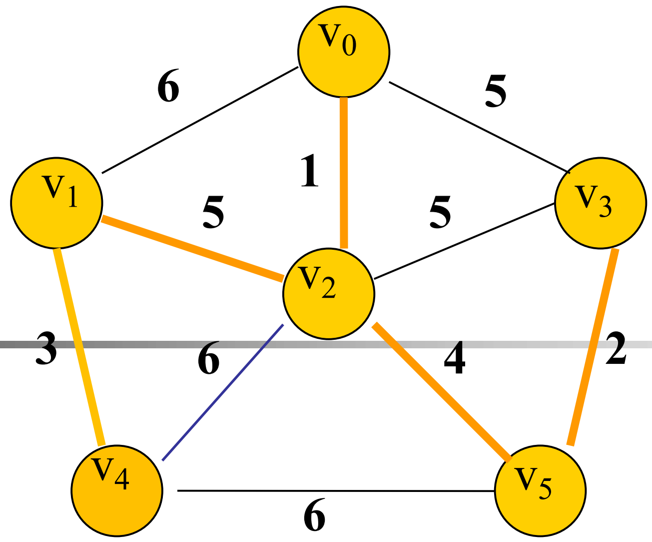
最小优先队列

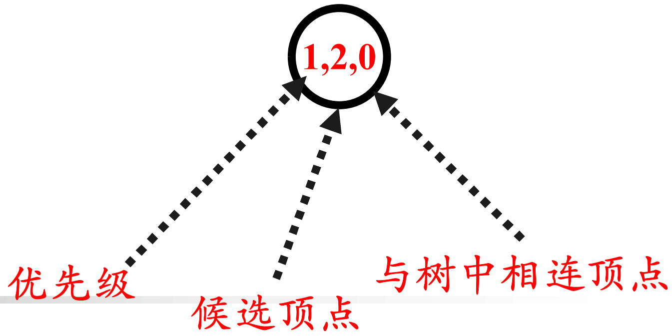




decreaseKey

最小优先队列



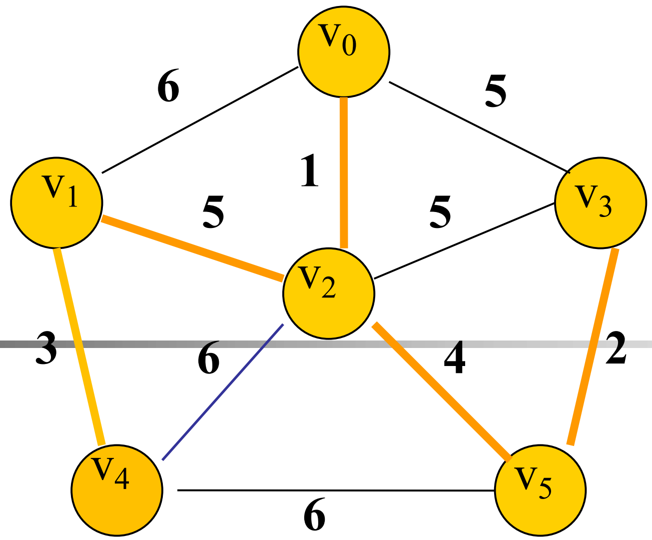


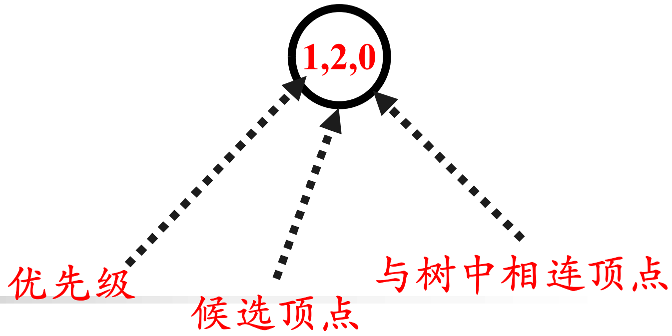
getMin取优先级最小的顶点及其相连的边加入生成树

deleteMin

3,4,1

最小优先队列





getMin取优先级最小的顶点及其相连的边加入生成树

deleteMin

改用优先队列存放fringe 顶点与生成树相连的最小边

最小优先队列

Prim算法

- MST性质(最小生成树性质):

令G=(V,E,W)为一个带权连通图，T为G的一生成树。对任一不在T中的边uv∈E，如果将uv加入T中会产生一回路，使得uv是回路中权值最大的边。那么树T具有MST性质。

Lemma 8.1 In a connected, weighted graph $G = (V, E, W)$, if T_1 and T_2 are two spanning trees that have the MST property, then they have the same total weight.

引理8.1 在带权连通图G=(V,E,W)中，如果2棵生成树T_1和T_2都具有MST性质，那么它们的总代价相同

引理8.1 在带权连通图G=(V,E,W)中，如果2棵生成树T_1和T_2都具有MST性质，那么它们的总代价相同

证明：数学归纳法----k：在T_1和T_2中不相同的边数。

(1) k=1： T_1和T_2有1条边不同，假设图G中$e_1 \neq e_2$, $e_1 \in E(T_1)$, $e_1 \notin E(T_2)$, $e_2 \in E(T_2)$, $e_2 \notin (T_1)$。

1.若$w(e_1)=w(e_2)$，则T_1和T_2总代价相同，引理成立。

2. 若$w(e_1) \neq w(e_2)$，不失一般性假设$w(e_1)<w(e_2)$.

➢由于$e_1=(u_1,v_1) \notin E(T_2)$, 在生成树$T_2$中必存在$u_1$和$v_1$路径$u_1=x_0x_1\cdots x_p=v_1$, $p\geqslant 2$。

➢那么将$e_1=(u_1,v_1)$加入T_2产生回路，由于T_2有MST性质，e_1是回路中权值最大的边，即$w(e_1)\geq w((x_i,x_{i+1}))$, $i=0,1,\cdots,p-1$,且(x_i,x_{i+1})中存在一条边不在T_1。假设(x_{i1},x_{i1+1})不在T_1：

若$e_2 \neq (x_{i1},x_{i1+1})$，与$T_1$和$T_2$有1条边不同矛盾

若$e_2=(x_{i1},x_{i1+1})$，则$w(e_1)\geq w((x_{i1},x_{i1+1}))=w(e_2)$,与$w(e_1)<w(e_2)$矛盾

➢所以k=1 T_1和T_2总代价相同

(2) 假设$1 \leqslant k<j$时2棵生成树T_1和T_2都具有MST性质，它们有j条边不同，但它们的总代价相同成立，那么对$k=j$进行证明。

设(u,v)是两棵树中不相同的边中权值最小的一条，不失一般性假设其在T_2而不在T_1中。u和v之间在T_1中存在一条路径$u=x_0x_1 \cdots x_p=v, p \geqslant 2$.由于$T_1$具有MST性质，该路径上所有边的权值不大于(u,v)的权值，且该路径上存在一条边(x_i, x_{i+1})不在T_2，$w((x_i, x_{i+1})) \leqslant w((u,v))$.

- $w((x_i, x_{i+1})) = w((u,v))$：$NT_1 = T_1 + (u,v) - (x_i, x_{i+1})$，$w(NT_1)=w(T_1)$，$NT_1$与$T_2$具有j-1条边不同，根据归纳假设$w(NT_1)=w(T_2)$，所以$w(T_1)=w(T_2)$。
- $w((x_i, x_{i+1}))<w((u,v))$的权值:与“(u,v)是两棵树中不相同的边中权值最小的一条”矛盾

定理8.2：在带权连通图 G=(V,E,W) 中，生成树T为最小生成树，当且仅当T具有MST性质

- → 树T为最小生成树,则其具有MST性质

➢ 对于u,v∈V，若边e=(u,v) ∉E(T),e∈E,由于T为生成树，是一个包含G中所有顶点的连通子图，则在树T中存在路径u= x_1, x_2, ⋯, x_r=v(r>2), 所以加入e后，形成回路(u=x_1, x_2, ⋯, x_r=v,u) (r>2);

➢ 假设回路中存在边**e1**= (x_i, x_{i+1}) (0<i<r),使得w(e)<w(e1)，则将e加入T,并从T中删除e1，得到一棵新的生成树T1，w(T1)=w(T)+w(e)-w(e1)<w(T),这与T为最小生成树矛盾。所以e的权值是回路边的权值最大的。

➢ 综上，树T为最小生成树,则其具有MST性质

- ← T具有MST性质,则树T为最小生成树

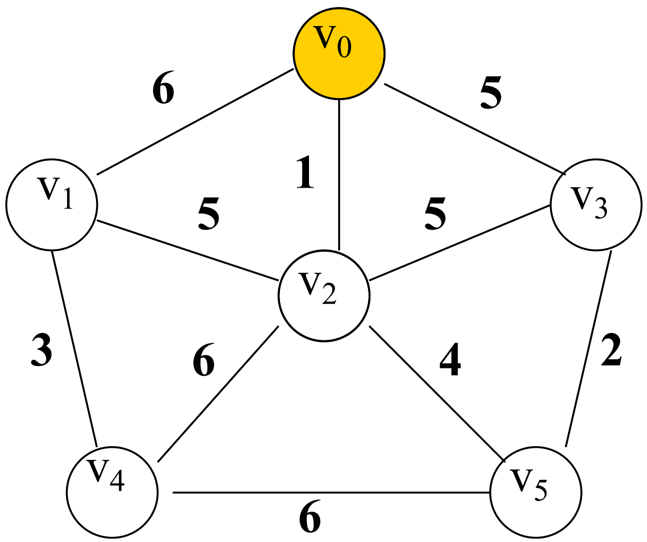
➢ 设T1为图G的最小生成树，则根据上述证明，T1具有MST性质。

➢ 由于T具有MST性质，根据引理8.1，图G具有MST性质的生成树代价相同，所以w(T)=w(T1), T为最小生成树.

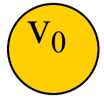
- 引理8.3：G=(V,E,W)为带权连通图，n=|V|,令T_k为prim算法构造的k个（k=1,2,⋯,n）顶点的树，G_k为图G由T_k中的顶点产生的导出子图。那么T_k在G_k中具有MST性质。
- 导出子图就是：一个顶点子集以及两个端点都在这个子集中的所有边构成的子图。

Lemma 8.3 Let $G = (V, E, W)$ be a connected, weighted graph with $n = |V|$; let T_k be the tree with k vertices constructed by Prim's algorithm, for $k = 1, \ldots, n$; and let G_k be the subgraph of G induced by the vertices of T_k (i.e., uv is an edge in G_k if it is an edge in G and both u and v are in T_k). Then T_k has the MST property in G_k.

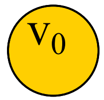
8.2 prim—最小生成树



T_1为

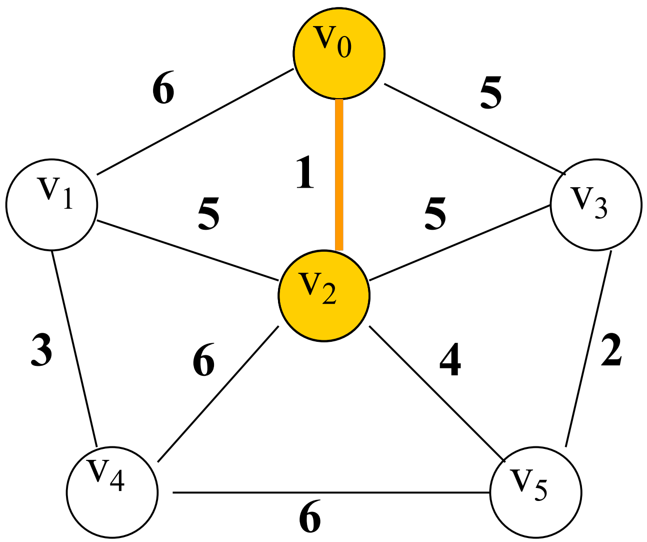


G_1为

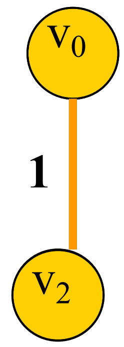


1. *tree* vertices: in the tree constructed so far,
2. *fringe* vertices: not in the tree, but adjacent to some vertex in the tree,
3. *unseen* vertices: all others.

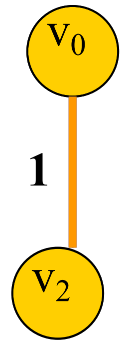
8.2 prim—最小生成树



T_2为

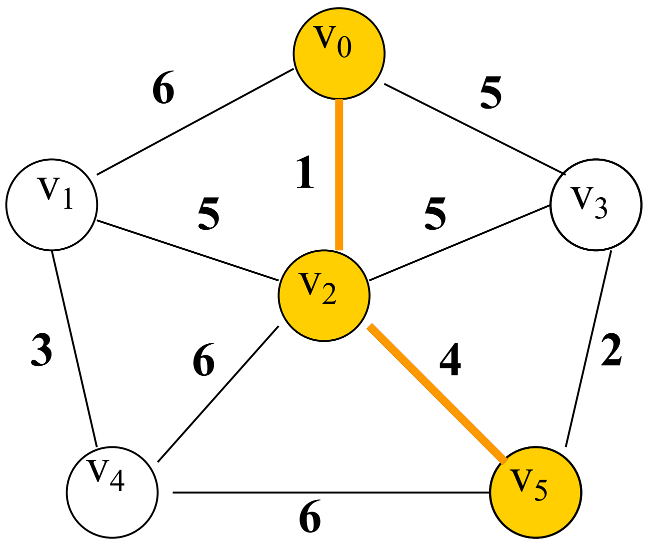


G_2为



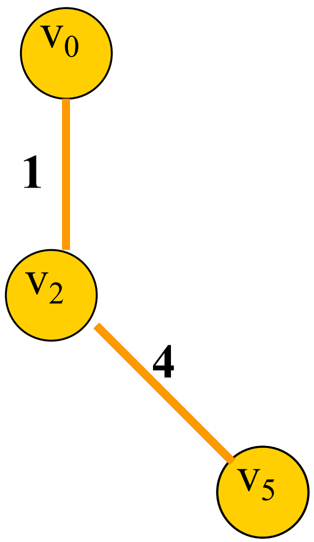
1. *tree* vertices: in the tree constructed so far,
2. *fringe* vertices: not in the tree, but adjacent to some vertex in the tree,
3. *unseen* vertices: all others.

8.2 prim—最小生成树



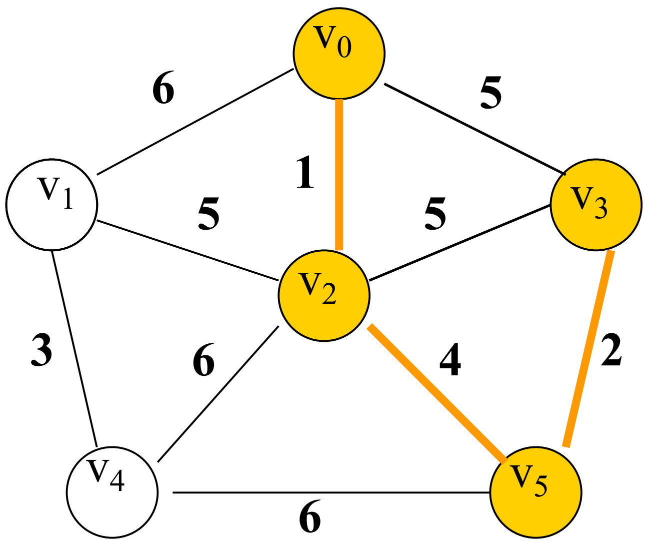
T_3为

G_3为

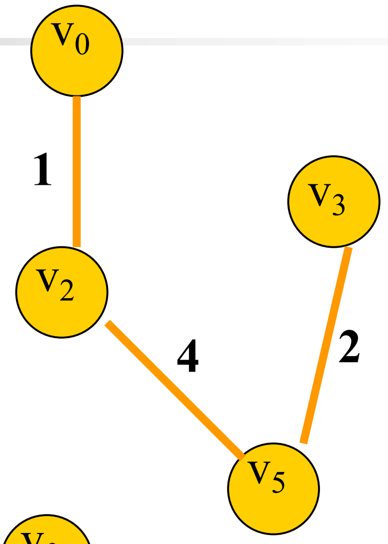


当一个问题的最优解包含其子问题的最优解时，称此问题具有最优子结构性质。问题的最优子结构性质是该问题可用动态规划算法或贪心算法求解的关键特征。

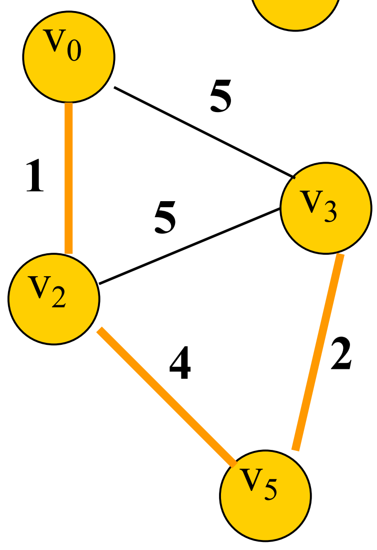
8.2 prim—最小生成树



T_4为



G_4为



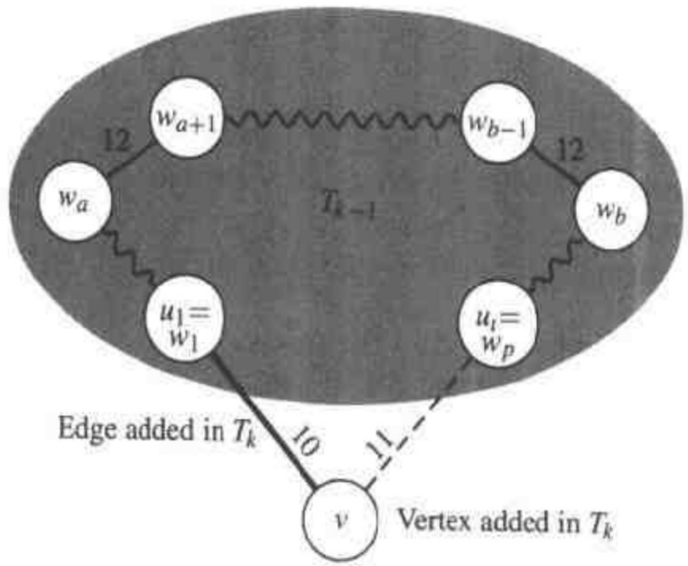
引理8.3：G=(V,E,W)为带权连通图，n=|V|，令T_k为prim算法构造的k个（k=1,2,⋯,n）顶点的树，G_k为图G由T_k中的顶点产生的导出子图。那么T_k在G_k中具有MST性质。

引理8.3：$G=(V,E,W)$为带权连通图，$n=|V|$,令T_k为prim算法构造的k个（$k=1,2,\cdots,n$）顶点的树，G_k为图G由T_k中的顶点产生的导出子图。那么T_k在G_k中具有MST性质。

证明：对k采用数学归纳法。

- (1) $k=1$, G_1和T_1都只有出发点s，所以T_1在G_1中具有MST性质。
- (2) $k>1$,假设$1\le k<j$时引理成立，T_k在G_k中具有MST性质。那么对$k=j$进行证明。假设prim算法加入的第j个结点是v，v与T_{j-1}相连的边是$u_1v,u_2v,\cdots,u_dv$,其中u_1v是权值最小的边。
- 考虑边$xy\in E(G_j)$，$xy\notin E(T_j)$
- (2.1) x,y均不是v ……．
- (2.2) x,y中一个是v

引理8.3：G=(V,E,W)为带权连通图，法构造的k个（k=1,2,⋯,n）顶点的树顶点产生的导出子图。那么T_k在G_k中



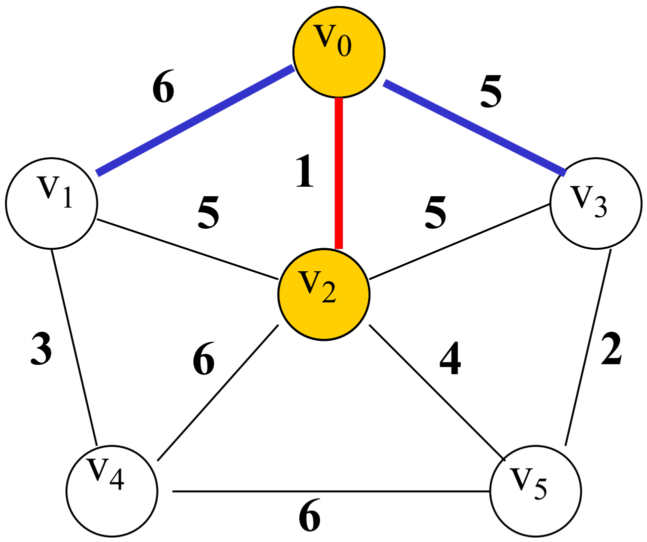
- (2.2) x,y中一个是v。若引理不成立成回路，其上存在边其权值大于xy
- 考虑路径$v,u_1=w_1,w_2,\cdots,w_p=u_i(p>1)$, T_{j-1}中.
- 令w_aw_{a+1}是该路径上第一条权值大于xy= u_iv 的权值的边，$w_{b-1}w_b$是该路径上最后一条权值大于xy= u_iv 的权值的边（a+1=b?）。
- 若w_a比w_b先加入生成树中：那么该路径上从w_1到w_a的边比w_aw_{a+1}或$w_{b-1}w_b$早加入树上，从而u_1v也比它们早加入树中，矛盾。
- 若w_b比w_a先加入树中：同理可得u_iv在T_{j-1}中，矛盾。
- 那么T_k在G_k中具有MST性质。

8.2 prim——最小生成树

- 定理8.4：prim算法构造的是最小生成树。
- 证明： 当图G的n个顶点均加入生成树时， $T_n=T$为prim算法的输出结果，$G_n=G$，根据引理8.3， $T_n=T$ 具有MST性质。根据定理8.2， $T_n=T$是图G的最小生成树。

prim算法实现

- 只需记录***fringe***结点与树的最小连边----candidate edge
- 优先队列----实现prim最小生成树算法
- 插入：将一个***fringe***结点插入队列，权值为其与树相连的最小边
- 查找
- 删除
- decreaseKey



基于优先队列实现prim算法

```
初始化一个空的优先队列PQ;
设定初始点s加入生成树;
for each edge (s,x)∈E(G)   insert (PQ,x,w(s,x))   ;
while    （PQ is not empty） {
         v=getMin(PQ); deleteMin(PQ);
         将v加入生成树;
         for each edge (v,x)∈E(G)
              if (xref[x]!=-1)
                 if (xref[x]==0)insert (PQ,x,w(v,x))   ;
                  else if（w(v,x)<x当前的权）decreaseKey(PQ,x,w(v,x));}
```

克鲁斯卡尔算法证明

- 定理8.2：在带权连通图 G=(V,E,W)中，树T为最小生成树，当且仅当T具有MST性质。
- 克鲁斯卡尔算法构造的生成树具有MST性质

克鲁斯卡尔算法证明

- 令G=(V,E,W)为一个带权连通图，设T为克鲁斯卡尔算法构造的生成树。e=(x,y)∈E但不在T中,T中存在路径P: $x=u_1,u_2,\cdots,u_p=y$, p>2 。
- 将e加入T会产生回路，假设且回路中存在权值比e大的边，不妨设e*是回路中权值最大的边, w(e)<w(e*)， $e^*= u_iu_{i+1}$, p>i>0 。
- 对于e和e*，克鲁斯卡尔算法执行时会先考察e=(x,y) ,因为算法没加入e，说明在加入e和e*前，就存在一条从x到y的不包含e*的路径Q。
- 因为e*是回路$(x=u_1,u_2,\cdots,u_p=y,x)$中权值最大的边, 在考察e*前, 回路上其他的边均已考察，则x和u_i在一个连通分量， y和u_{i+1}在一个连通分量， x和y在一个连通分量。所以u_{i+1}和u_i在一个连通分量，算法不会加入e*，矛盾。