算法设计与分析—进阶篇

第四讲贪心法与拟阵

哈尔滨工业大学
王宏志
wangzh@hit.edu.cn
http://homepage.hit.edu.cn/pages/wang/

本讲内容

4.1 最小生成树算法

4.2 拟阵概述

4.3 从拟阵看任务安排问题

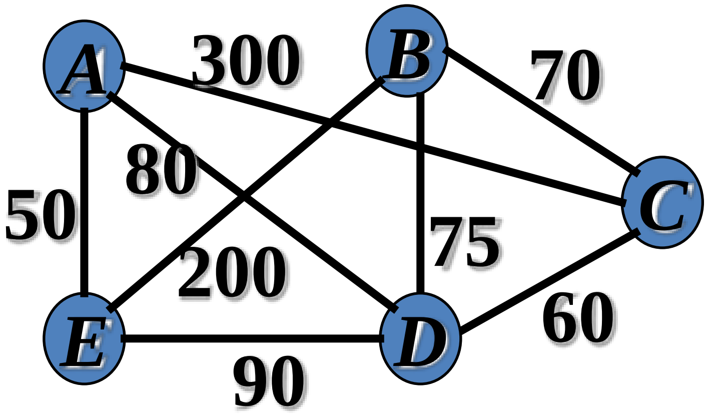
问题的定义

- 生成树
  - 设$G=(V, E)$是一个边加权无向连通图. G的生成树是无向树$S=(V, T)$, $T\subseteq E$, 以下用T表示S.
  - 如果 $W: E\rightarrow\{实数\}$是G的权函数, T的权值定义为$W(T)=\sum_{(u,v)\in T}W(u,v)$.
- 最小生成树
  - G的最小生成树是$W(T)$最小的G之生成树.
- 问题的定义

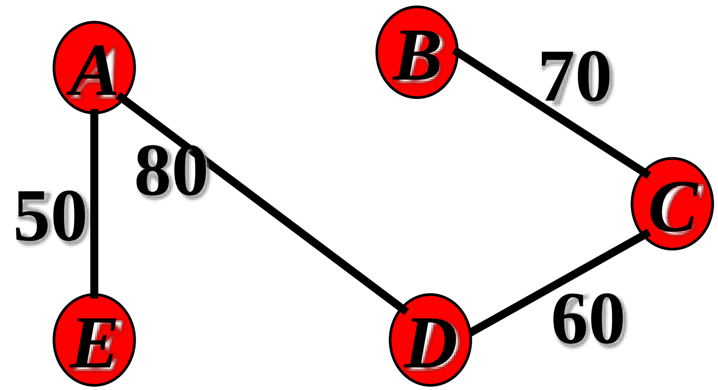
  输入: 无向连通图$G=(V, E)$, 权函数W

  输出: G的最小生成树

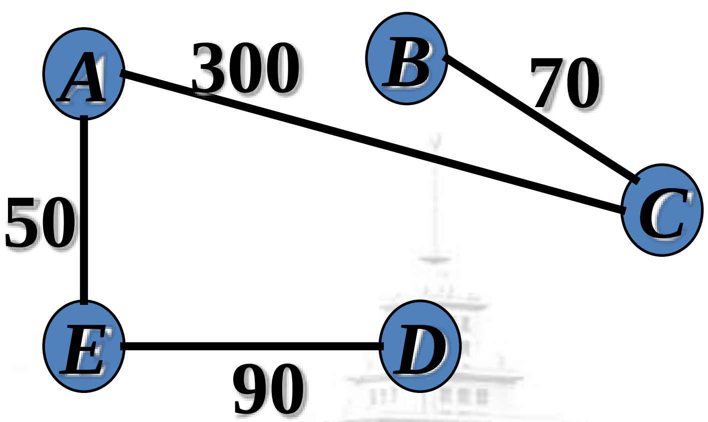
- **实例**



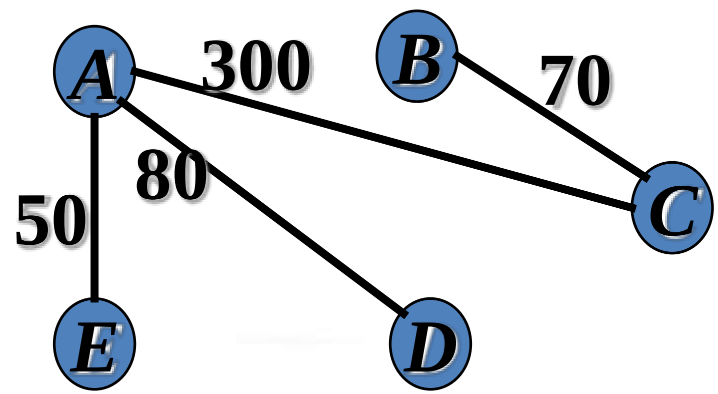
A
B
C
D
E
300
70
50
80
200
75
60
90



A
B
C
D
E
70
80
50
60

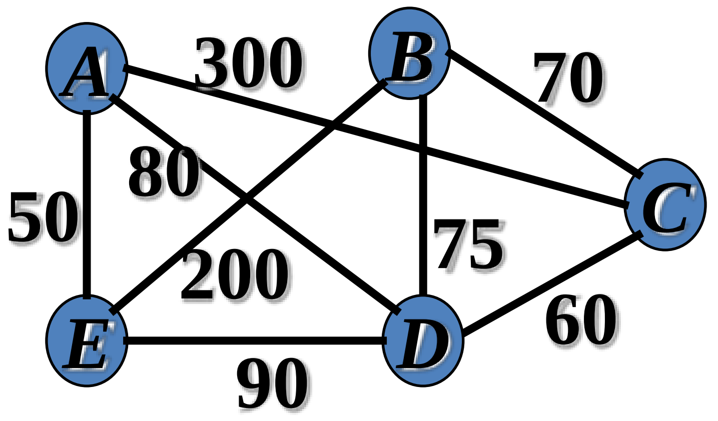


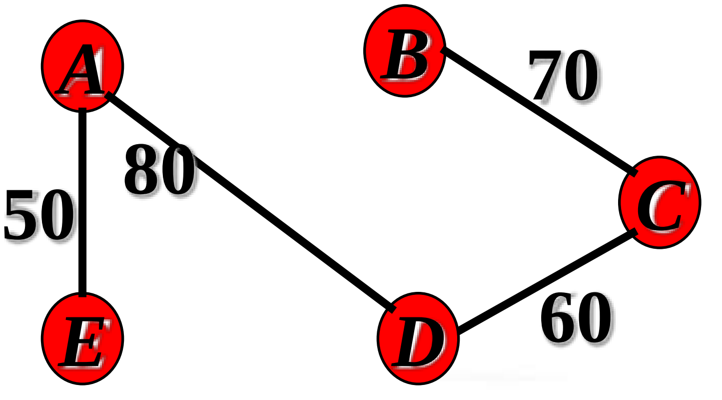
A
B
C
D
E
300
70
50
90



A
B
C
D
E
300
70
80
50

• 算法思想





Kruskal算法

MST-Kruskal(*G*,*W*)

A=∅;

For ∀*v*∈*V[G]* Do

Make-Set(*v*); /*

按照*W*值的递增顺序排序*E[G]*;

For ∀(*u*, *v*) ∈*E[G]* (按*W*值的递增顺序) Do

If Find-Set(*u*)≠Find-Set(*v*)

Then A=A∪{(u, v)}; Union(*u*, *v*);

Return *A*

算法复杂性

- 令$n=|V|$, $m=|E|$
- 第4步需要时间: $O(mlogm)$
- 第2-3步执行$O(n)$个*Make-Set*操作

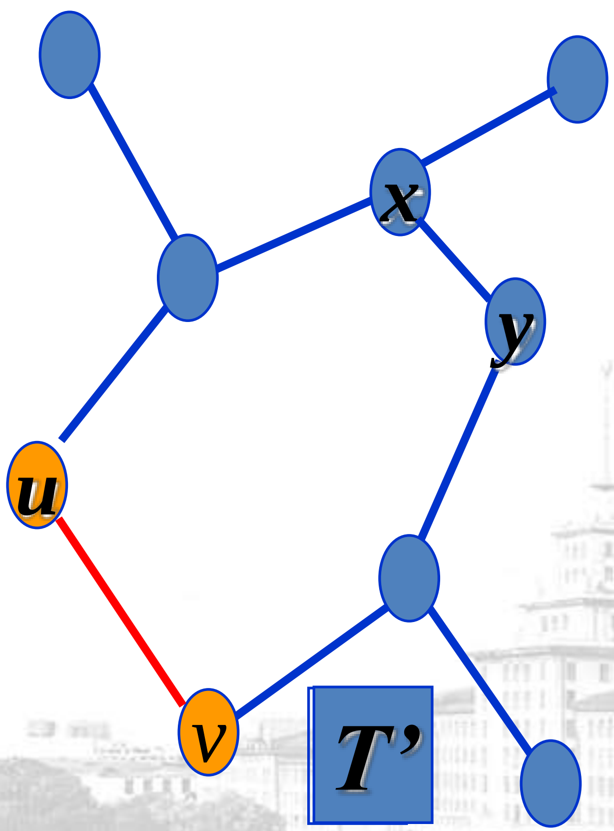
  第5-8步执行$O(m)$个Find-Set和Union操作

  需要时间: $O((n+m)\alpha(n))$
- $m \geq n-1$(因为G连通),

  $\alpha(n)=logn=logm$
- 总时间复杂性：$O(mlogm)$

贪心选择性

定理1. 设uv是G中权值最小的边，则必有一棵最小生成树包含边uv.



证明：设T是G的一棵MST

若$uv \in T$, 结论成立；

否则，如右图所示

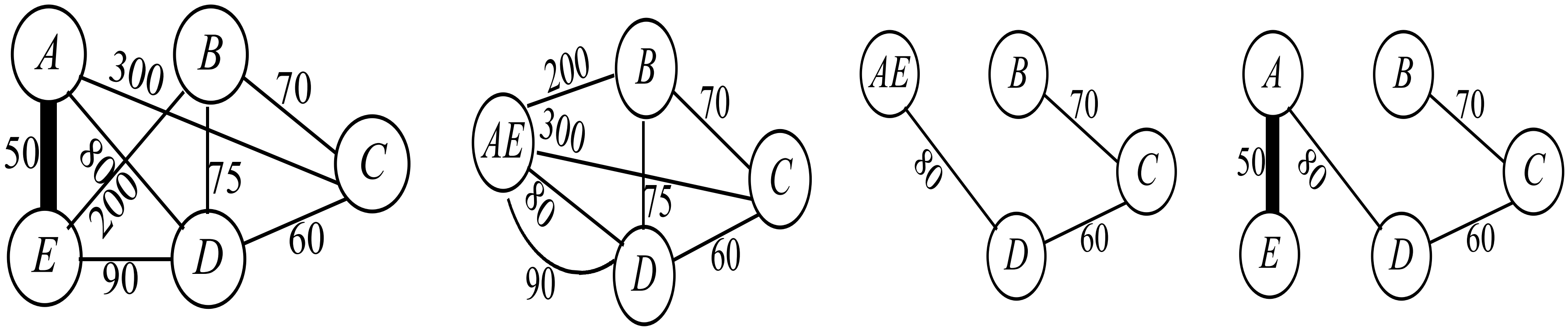
在T中添加uv边，产生环

删除环中不同于uv的权值最小的边xy,得到T'。

$w(T')=w(T)-w(xy)+w(uv)$

$\leq w(T)$

优化子结构



收缩图*G*的边*uv*—***G*•*uv***

- 用新顶点C_{uv}代替边*uv*
- 将*G*中原来与*u*或*v*关联的边与C_{uv}关联
- 删除C_{uv}到其自身的边

上述操作的逆操作称为扩张

定理1.给定加权无向连通图$G=(V,E)$,权值函数为 $W:E\rightarrow R$, $uv\in E$是G中权值最小的边。设T是G的包含uv的一棵最小生成树，则$T\cdot uv$是$G.uv$的一棵最小生成树.

证明. 由于$T\cdot uv$是不含回路的连通图且包含了$G\cdot uv$的所有顶点，因此，$T\cdot uv$是$G\cdot uv$的一棵生成树。下面证明$T\cdot uv$是$G\cdot uv$的代价最小的生成树。

若不然，存在$G\cdot uv$的生成树T'使得$W(T')<W(T\cdot uv)$。显然，T'中包含顶点C_{uv}且是连通的，因此$T''=T'\circ C_{uv}$包含G的所有顶点且不含回路，故T''是G的一棵生成树。但,$W(T'')= W(T')+W(uv) <W(T\cdot uv)+W(uv)=W(T)$,这与$T$是$G$的最小生成树矛盾。

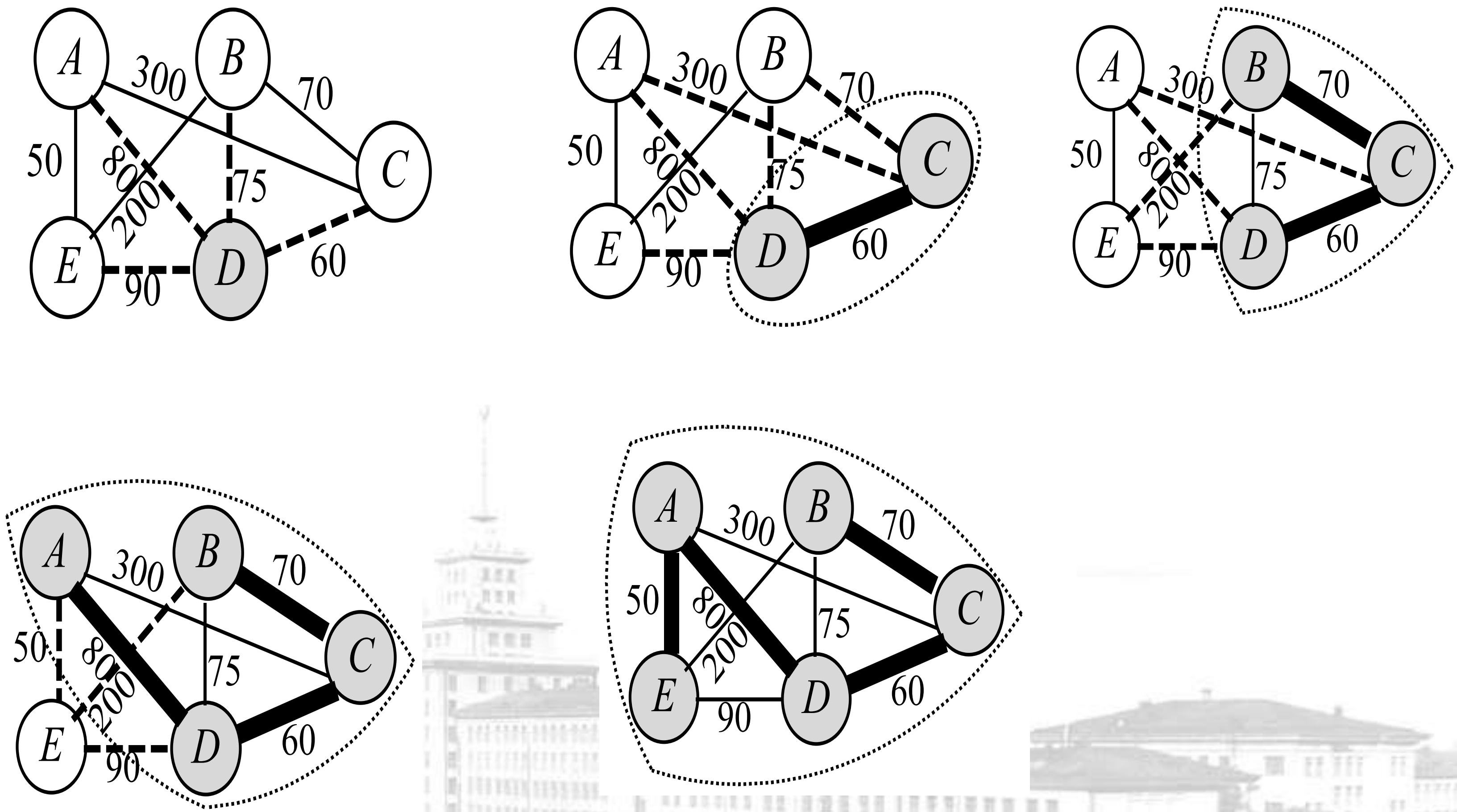
算法正确性

定理2. MST-Kruskal(G,W)算法能够产生图G的最小生成树.

证. 因为算法按照贪心选择性进行局部优化选择.

Prim算法

- **算法思想**



算法描述(1)

MST-Prim(G,W,r)

Input 连通图G，权值函数W，树根r

Output G的一棵以r为根的生成树

1. $C \leftarrow \{r\}$，$T \leftarrow \varnothing$;
2. 建堆Q维护C与V-C之间的边 　　log E
3. While $C \neq V$ do
4. 　　$uv \leftarrow$ Extract_Min(Q) 　　//$u \in C, v \in V-C$
5. 　　$C \leftarrow C \cup \{v\}$; 　　$T \leftarrow T \cup \{uv\}$;
6. 　　for $\forall x \in$ Adj[v] do 　　2E遍
7. 　　　　if $x \in C$ then 将vx从Q中删除 　　log E
8. 　　　　Else 　　将vx插入Q
9. Return T

算法描述(2)

```
MST-Prim(G,W,r)
Input   连通图G，权值函数W，树根r
Output G的一棵以r为根的生成树
For  ∀ v∈V[G]   Do
        key[v]←+∞
        π[v] ←null
key[r] ←0
Q ←V[G]
While Q≠∅ do
        u ←Extract_Min(Q)                                   log V
for ∀v∈Adj[u]  do                                           2E遍
            if v∈Q 且 w(u,v)<key[v]   then                   常数
                        π[v]     ←u                           时间
                        key[v] ←w(u,v)          //更新信息    logV
Return  A={(v, π[v])| v ∈V[G] -r}
```

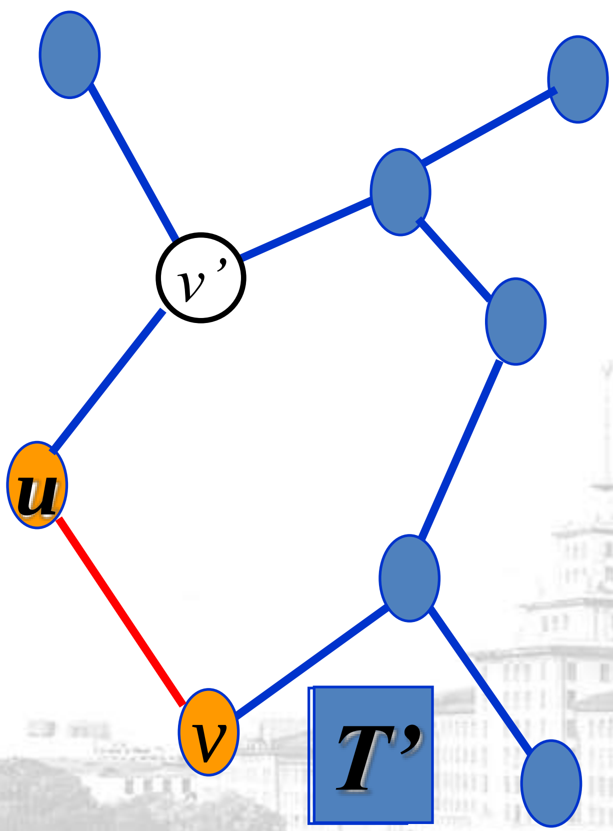
算法复杂性

假设用最小堆实现*Q*

总的时间开销为O(VlogV+ElogV)=O(ElogV)

贪心选择性

定理1. 设uv是G中与顶点u关联的权值最小的边，则必有一棵最小生成树包含边uv.



证明：设T是G的一棵MST

若$uv \in T$, 结论成立；

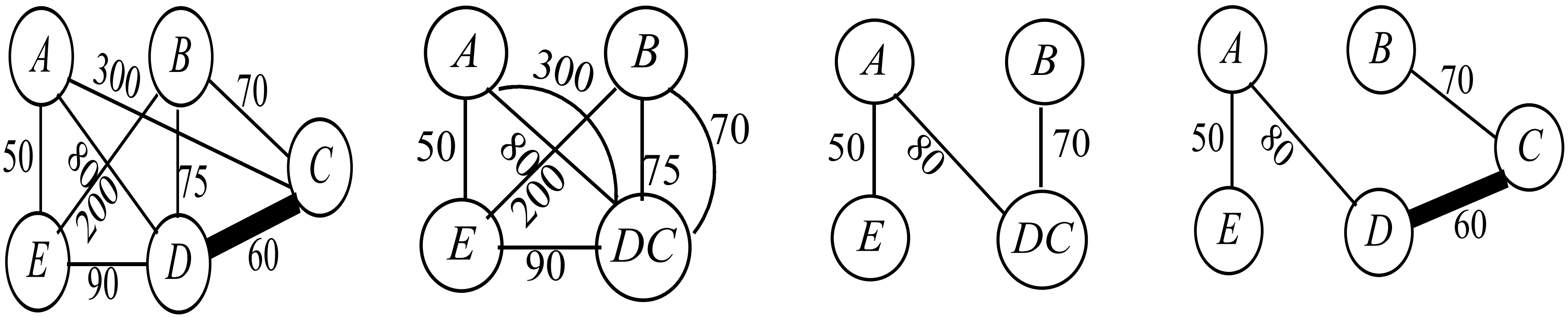
否则，如右图所示

在T中添加uv边，产生环，环中顶点u的度为2，即存在$uv' \in T$.

删除环中边uv',得到T'。

$w(T')=w(T)-w(xy)+w(uv) \le w(T)$

优化子结构



收缩图*G*的边*uv*—*G•uv*

- 用新顶点C_{uv}代替边*uv*
- 将*G*中原来与*u*或*v*关联的边与C_{uv}关联
- 删除C_{uv}到其自身的边

上述操作的逆操作称为扩张

定理1.给定加权无向连通图$G=(V,E)$,权值函数为 $W:E\rightarrow R$, $uv\in E$是G中顶点u关联的权值最小的边。设T是G的包含uv的一棵最小生成树，则$T\cdot uv$是$G.uv$的一棵最小生成树.

证明. 同Kruskal算法优化子结构的证明。

算法正确性

定理2. MST-Prim(G,W)算法能够产生图G的最小生成树.

证. 因为算法按照贪心选择性进行局部优化选择.