图论与代数结构

回路割集矩阵与最优树

崔 勇
清华大学计算机系
网络技术研究所

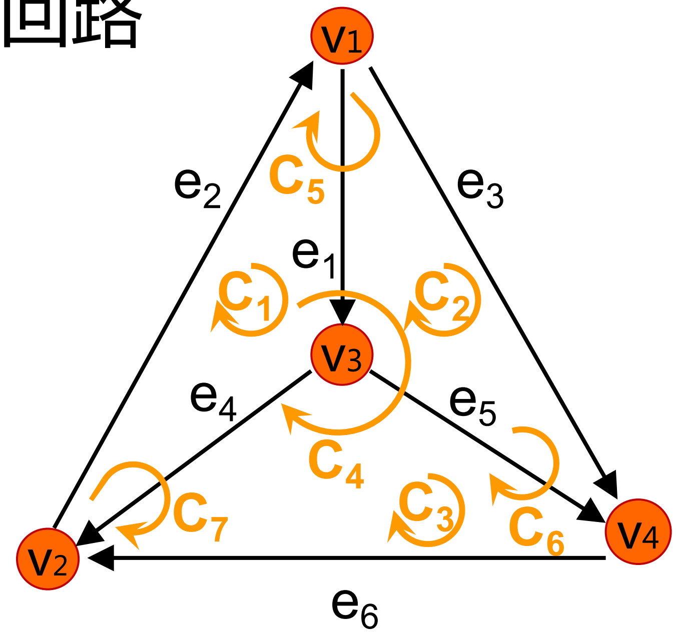
- 本节课重点
  - 回路矩阵与割集矩阵计算
  - Huffman树算法
  - 最短树算法

第三章 树

- 树的有关定义
- 基本关联矩阵及其性质
- 支撑树的计数
- 回路矩阵与割集矩阵
- Huffman树
- 最短树

回路矩阵

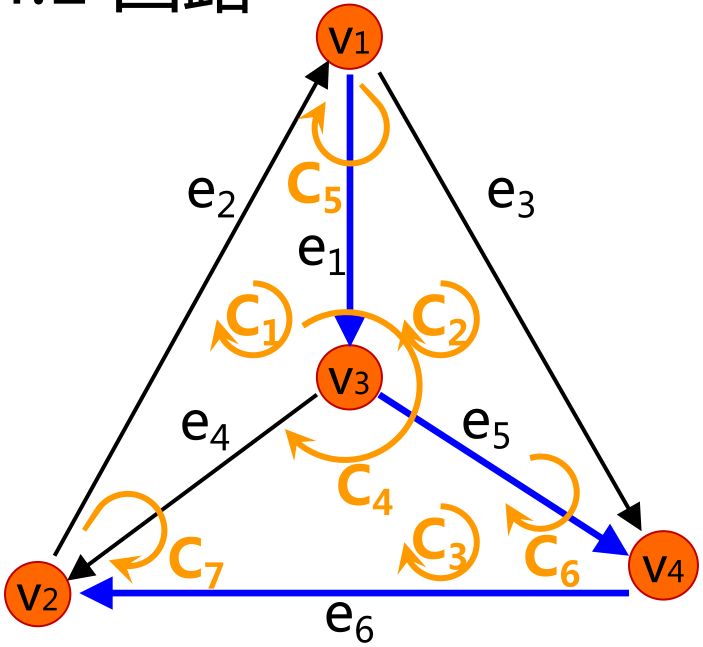
- 例3.4.1 回路



考虑每条余树边是否使用……

回路矩阵(2)

- 例3.4.1 回路



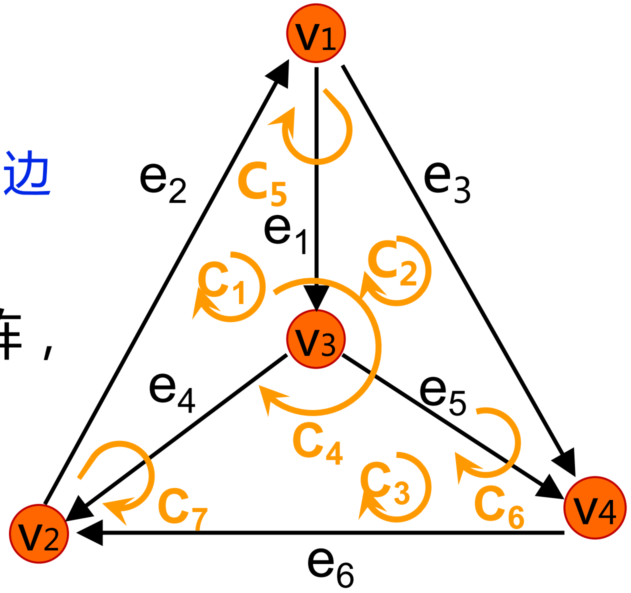
$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array}\begin{array}{c}\begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array}\\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & -1 & -1 & 1 \\ 0 & 0 & -1 & 1 & 0 & 0 \end{bmatrix}\end{array}$$

最多可能包含$2^{m-n+1}-1$个不同的初级回路

回路矩阵(3)

- 回路的代数表示
  - 给定C的一参考方向
    - 该回路的边若与参考方向一致就称为正向边
    - 否则就称为反向边
  - 有向连通图G的全部初级回路构成的矩阵，称为G的完全回路矩阵，记为C_e

$$C_{ij}=\begin{cases}1, & e_j\in C_i\text{且与回路}C_i\text{方向一致。}\\ -1, & e_j\in C_i\text{且与回路}C_i\text{方向相反。}\\ 0, & \text{其他}\end{cases}$$



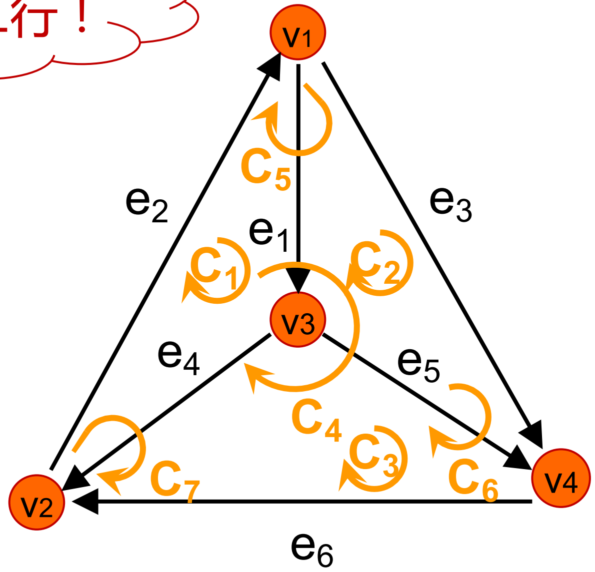
回路矩阵(4)

- 例3.4.1
  - 完全回路矩阵

$$C_e = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & -1 & 0 \\ -1 & 0 & 1 & -1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 \end{bmatrix}$$

  - 这些回路是否是独立的？

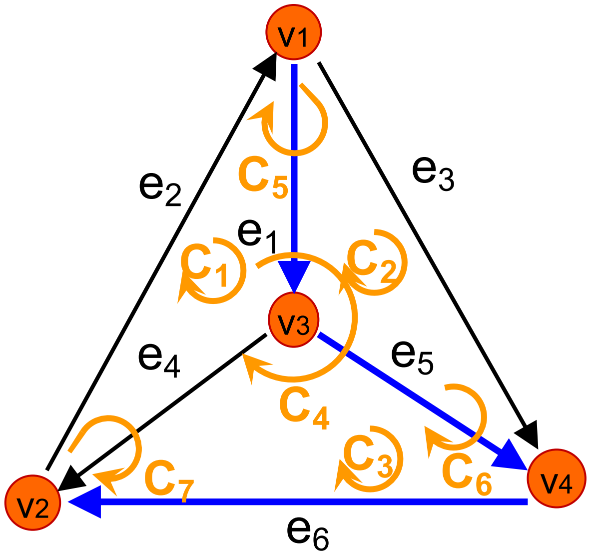
2^{m-n+1}-1行！



$C_1 \oplus C_2 = C_5 \qquad C_1 \oplus C_3 = C_7$

回路矩阵(5)

- 基本回路的概念
  - 设T是图G=(V,E)的一棵支撑树
  - 对任一余树边e∈E(G)- E(T), T+e都构成一个唯一回路C
  - 当有向图G=(V,E)的树T确定后，每条余树边e与T的子集所对应的回路称为基本回路
  - 基本回路的方向与余树边e的方向一致
- 基本回路矩阵C_f
  - 由全部基本回路构成的矩阵，称为G的基本回路矩阵，记为$C_{f\ (m-n+1)*m}$

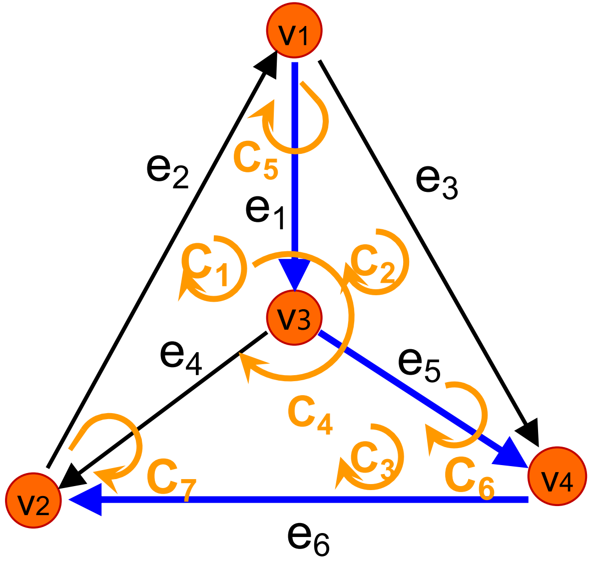


回路矩阵(6)

- 例3.4.2
  - 给定图的一棵树T={e_1,e_5,e_6} ,则其基本回路矩阵是

$$
\begin{array}{cc}
C_f = & \left[\begin{array}{c|ccc|cc} 1 & 1 & 0 & 0 & 1 & 1 \\ -1 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & -1 & -1 \end{array}\right] \\
 & \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array}
\end{array}
$$

余 树 边



  - 显然基本回路矩阵中每个回路都是独立的
  - 因此ran C_f=m-n+1

回路矩阵(8)

- 定理3.4.1 (引理)
  - 有向连通图G=(V,E)的关联矩阵$B_{n\times m}$与完全回路矩阵Ce $_{(\sim)\times m}$的边次序一致时，恒有：

$$BC_e^T = 0$$

B: 结点和边的关系
C: 回路和边的关系

- 证明：
  - 设 $D=BC_e^T, d_{ij} = \sum_{k=1}^{m} b_{ik} \bullet c_{jk}$
  - 其中b_{ik}是结点v_i与边e_k的关联状况
  - c_{jk}是回路 c_j与边e_k的相关情况
  - d_{ij}不为0：给定v_i和c_j，存在边e_k与v_i和c_j都关联

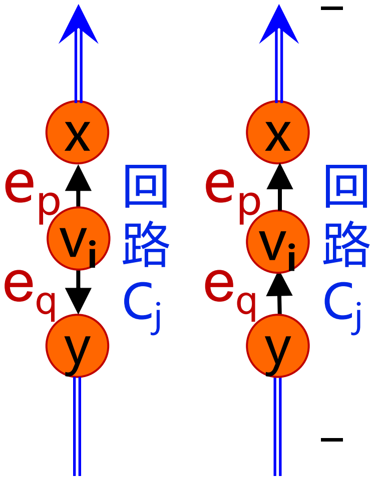
回路矩阵(9)

证(续)

$$D=BC_e^T, d_{ij}=\sum_{k=1}^{m} b_{ik}\cdot c_{jk}$$

– 考虑回路C_j与结点 v_i的相处，只有两种可能：

– (1) C_j不经过v_i，则与v_i关联的任一边都不是C_j中的边，所以$b_{ik}=0$，即$d_{ij}=0$



– (2) C_j经过v_i, 则必定经过与v_i关联的2条边 e_p和e_q

- 如果e_p和e_q在C_j中方向相反，对 v_i它们却是同进同出的，因此，对 e_p和e_q有b不变而c相反，即$d_{ij}=0$
- 若e_p和e_q在C_j中方向一致，则对v_i来说它们是一进一出的，因此，对 e_p和e_q有c不变而b相反，即$d_{ij}=0$

$\boxed{BC^T=0}$

– 由于 d_{ij}的任意性，故定理得证 $BC_e^T=0$

回路矩阵(7')

- 呼唤定理
  - 有向连通图G=(V,E)的完全回路矩阵C_e的秩是m-n+1

如何证明呢？

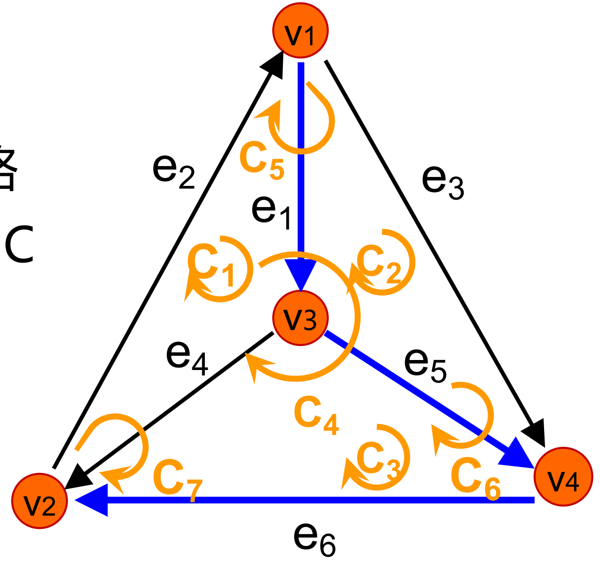
- 苦思冥想不得其解☹
- 找找数学系的朋友（$BC_e^T=0$）：
  - Sylvester 定理指出，两个矩阵$A_{n\times s}$, $B_{s\times m}$，如果 AB=0，则ran A+ran B ≤s

回路矩阵(10)

- 定理3.4.2
  - 有向连通图G=(V,E)的完全回路矩阵C_e的秩是m-n+1
- 证明：
  - 由于基本回路矩阵C_f 是完全回路矩阵C_e 的子阵且 ran C_f =m-n+1, 故ran C_e ≥m-n+1
  - 现证ran C_e ≤m-n+1
  - Sylvester定理指出：两个矩阵$A_{n\times s}$, $B_{s\times m}$，如果 AB=0，则ran A+ran B ≤s
  - 由定理3.4.1$BC_e^T=0$, 得到 ran B+ran C_e ≤m, 关联矩阵B有ran B=n-1
  - 因此 ran C_e ≤m-n+1

回路矩阵(11)

- 定义3.4.3：回路矩阵
  - 由连通图G中m-n+1个互相独立的回路组成的矩阵，称为G的回路矩阵，记为C
- 回路矩阵的简单性质：
  1. 基本回路矩阵 C_f 是回路矩阵
  2. $BC^T=0$, (其中B和C的边次序一致)
  3. $C=PC_f$，其中P是非奇异(即行列式不为0)的方阵，C与基本回路矩阵 C_f 的边次序一致

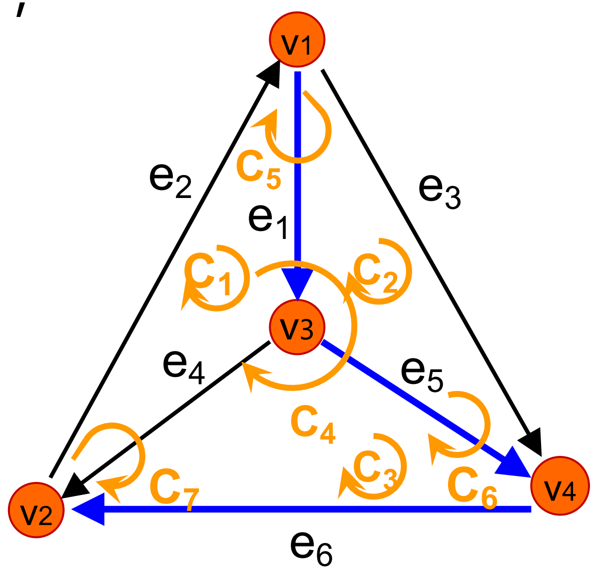


回路矩阵(12)

- 例3.4.2:给定图的一棵树$T=\{e_1,e_5,e_6\}$，则其基本回路矩阵是

$$C_f = \begin{array}{c} \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 \\ -1 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & -1 & -1 \end{bmatrix} \\ \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \end{array}$$

  - 每条余树边用且仅用1次
  - $\text{ran}\ C_e = \text{ran}\ C_f = m-n+1$



回路矩阵(13)

- 每条余树边用且仅用1次
  - 将行、列分别进行交换，使树枝边放在后，余树边放在前且次序与它所构成的回路一致，就可以写成分块矩阵形式

$$C_f = \begin{array}{c} \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 \\ -1 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & -1 & -1 \end{bmatrix} \\ \begin{matrix} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{matrix} \end{array} \qquad C_f = \begin{array}{c} \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & -1 & -1 & 0 \\ 0 & 0 & 1 & 0 & -1 & -1 \end{array}\right] \\ \begin{matrix} e_2 & e_3 & e_4 & e_1 & e_5 & e_6 \end{matrix} \\ \underbrace{\qquad\quad}_{\text{余树边}} \quad \underbrace{\qquad\quad}_{\text{树枝边}} \end{array}$$

  - 亦即 $C_f = \begin{pmatrix} I & C_{f_{12}} \end{pmatrix}$
  - 其中 $C_{f_{12}}$ 是树枝边所对应的子阵

$\boxed{BC^T=0\text{：呼唤定理}}$

回路矩阵(14)

- 定理3.4.4
  - 若有向连通图G=(V,E)的基本关联矩阵B_k和基本回路矩阵C_f的边次序一致，并设$C_f=(I\ C_{f12})$，$B_k=(B_{11}\ B_{12})$，分别对应余树边和树枝边，则

余树边 树枝边

$$C_{f_{12}} = -B_{11}^{T}B_{12}^{-T}$$

$$C_f = \begin{array}{c} \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & -1 & -1 & 0 \\ 0 & 0 & 1 & 0 & -1 & -1 \end{array}\right] \\ \begin{array}{cccccc} e_2 & e_3 & e_4 & e_1 & e_5 & e_6 \end{array} \end{array}$$

余树边 树枝边

- 定理3.4.3
  - 连通图G的回路矩阵C的任一(m-n+1)阶子阵行列式非零,当且仅当这些列对应于G的某一棵余树
  - 即去掉的(n-1)条边不含基本回路

回路矩阵(15)

- 例3.4.3：已知图3.11基本关联矩阵

$$
\begin{array}{c}
B_4 = \begin{bmatrix} 1 & -1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & -1 \\ -1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \\
\quad\quad\quad e_1 \;\; e_2 \;\; e_3 \;\; e_4 \;\; e_5 \;\; e_6
\end{array}
$$

- 其中e_1,e_5,e_6所对应的矩阵行列式非零
- 求基本回路矩阵C_f

回路矩阵(16)

- 解：由e_1,e_5,e_6可构成G的一棵树，对B_4进行列变换，得到

$$B_4 = \begin{array}{c} \begin{bmatrix} -1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & -1 \\ 0 & 0 & 1 & -1 & 1 & 0 \end{bmatrix} \\ \begin{matrix} e_2 & e_3 & e_4 & e_1 & e_5 & e_6 \end{matrix} \end{array} = \begin{pmatrix} B_{11} & B_{12} \end{pmatrix}$$

余树边　树枝边

其中 $B_{11} = \begin{bmatrix} -1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 1 \end{bmatrix}$ $B_{12} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ -1 & 1 & 0 \end{bmatrix}$ $B_{12}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$

$$C_{f_{12}} = -B_{11}^{T} B_{12}^{-T}$$

回路矩阵(17)

- 基本回路矩阵的右子阵

$$C_{f_{12}} = -B_{11}^{T} B_{12}^{-T} = \begin{bmatrix} 1 & 1 & 1 \\ -1 & -1 & 0 \\ 0 & -1 & -1 \end{bmatrix}$$

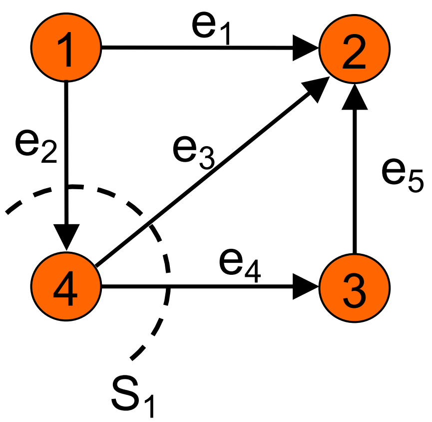
- 基本回路矩阵

$$\begin{array}{c} C_f = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & -1 & -1 & 0 \\ 0 & 0 & 1 & 0 & -1 & -1 \end{bmatrix} \\ \begin{array}{cccccc} e_2 & e_3 & e_4 & e_1 & e_5 & e_6 \end{array} \end{array}$$

余树边　　树枝边

割集矩阵

- 定义3.4.4 割集
  - 设S是有向图G=(V,E)的边子集，若同时满足：
    1. G'=(V,E-S) 比G的连通支数多1
    2. 对任意$S' \subset S$，G与G"=(V,E-S')的连通支数相同
  - 则称S是G的一个割集



$S_1=\{e_2,e_3,e_4\}$是割集

单选题 1分

设置

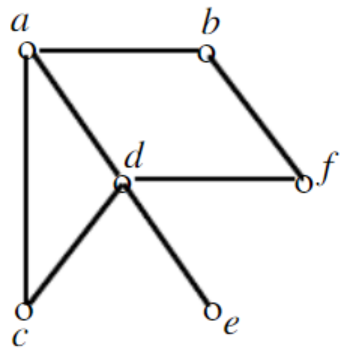
下图中关于割边割集的说法，正确的是：

A {ac, ad}是割集

B {ab, df, de}是割集

C {ac, ad, bf}是割集

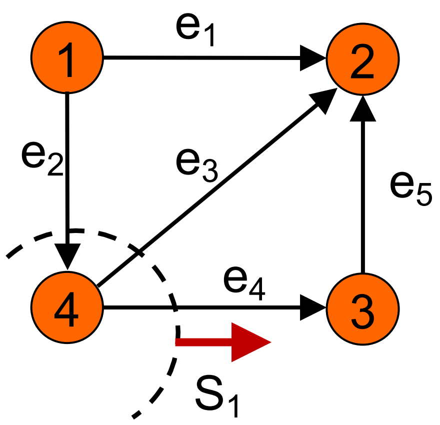
D {bf, df, de}是割集



提交

割集矩阵(2)

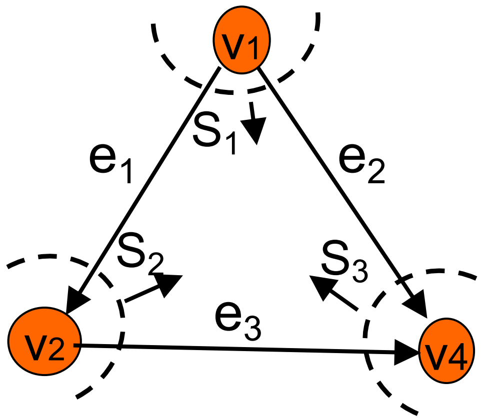
- 割集的方向
  - 给S确定一个方向，则S中每条边e与S 同向或反向
- 例3.4.4
  - e_2与S_1方向相反, e_3、e_4与S_1方向相同



$$
B_4 = \begin{array}{c}
\begin{bmatrix} 1 & -1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & -1 \\ -1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \\
\begin{matrix} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{matrix}
\end{array}
\quad
C_f = \begin{array}{c}
\begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & -1 & -1 & 0 \\ 0 & 0 & 1 & 0 & -1 & -1 \end{bmatrix} \\
\begin{matrix} e_2 & e_3 & e_4 & e_1 & e_5 & e_6 \end{matrix}
\end{array}
$$

割集矩阵(3)

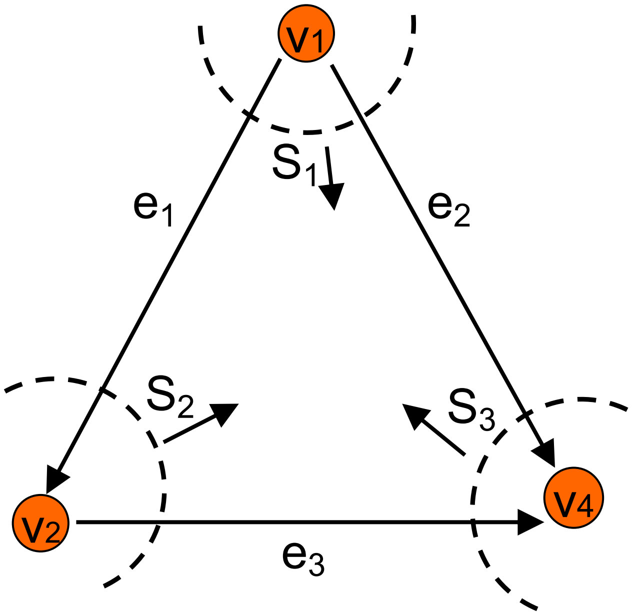
- 定义3.4.5
  - 有向连通图G的全部割集组成的矩阵，称为完全割集矩阵。记作S_e，其元素 ⇨ $S_{ij}=\begin{cases}1, & e_j\text{在}S_i\text{中且方向一致}\\ -1, & e_j\text{在}S_i\text{中且方向相反}\\ 0, & \text{其他}\end{cases}$
- 例3.4.5 求左图的完全割集矩阵



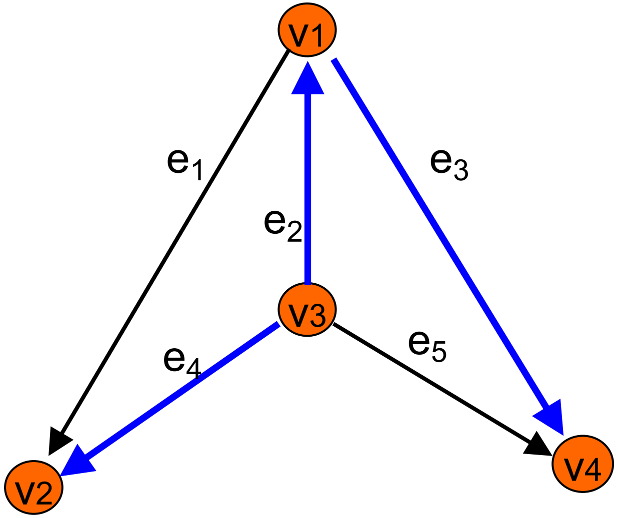
$$\Rightarrow \quad S_e=\begin{array}{c} \begin{matrix} e_1 & e_2 & e_3 \end{matrix} \\ \begin{pmatrix} 1 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & -1 \end{pmatrix} \end{array} \begin{matrix} S_1 \\ S_2 \\ S_3 \end{matrix}$$

割集矩阵(4)

- 例3.4.5 求左图的完全割集矩阵

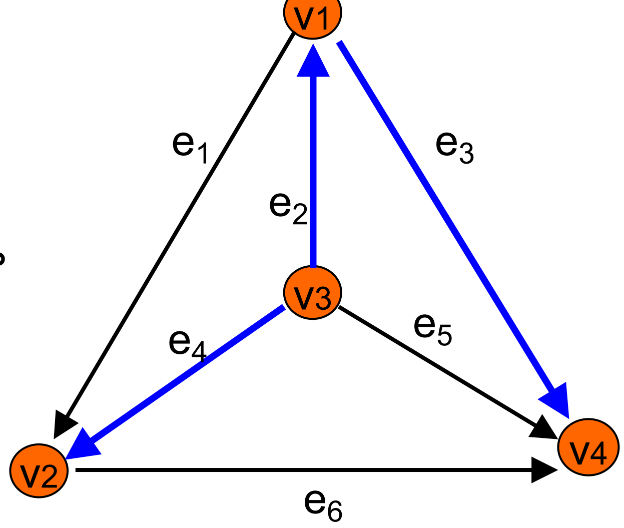


树枝边
v.s.
余树边



割集矩阵(5)

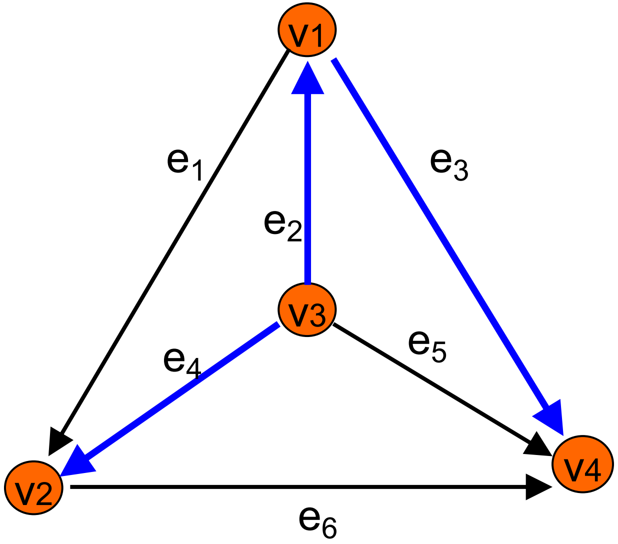
- 定义3.4.6
  - 设T是连通图G的一棵树，e_i是一个树枝。
  - 对树枝边e_i存在G的割集S_i，S_i只包括一条树枝边e_i及某些 余树边，且与e_i方向一致。
  - 这时S_i为G的对应树T的一个基本割集。
- 定义3.4.7
  - 给定有向连通图G的一棵树T，则由全部基本割集组成的矩阵称为基本割集矩阵S_f
  - ran S_f = n - 1



割集矩阵(6)

$$S_f = \begin{array}{c} \begin{bmatrix} -1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & -1 \end{bmatrix} \\ \begin{matrix} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{matrix} \end{array}$$

$$S_f = \begin{array}{c} \begin{bmatrix} -1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & -1 & 0 & 0 & 1 \end{bmatrix} \\ \begin{matrix} e_1 & e_5 & e_6 & e_2 & e_3 & e_4 \end{matrix} \end{array}$$



余树边 & 树枝边

基本割集矩阵满秩

割集矩阵(7)

- 定理3.4.5
  - 当有向连通图G的完全回路矩阵C_e和完全割集矩阵S_e的边次序一致时，有 $S_e C_e^T = 0$

考虑：$BC^T=0$
回路和割集必重复偶数条边

- 定理3.4.6
  - 连通图G的完全割集矩阵S_e的秩是n-1
    - 基本割集矩阵S_f是S_e的子矩阵，而ran S_f = n - 1，因此ran $S_e \geq n-1$
    - 由定理3.4.5及Sylvester定理
    - ran S_e +ran $C_e \leq m$，而ran C_e = m-n+1，故ran $S_e \leq n-1$

割集矩阵(8)

- 定义3.4.8：割集矩阵S
  - 连通图G的n-1个互相独立的割集构成的矩阵成为G的割集矩阵，记为S。
- 割集矩阵S有以下性质
  - 基本割集矩阵S_f是割集矩阵
  - $SC^T=0$，S和C的边次序一致
  - $S_f = PS$，其中P是非奇异方阵，S与S_f的边次序一致

$$S_f = \begin{bmatrix} -1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & -1 & 0 & 0 & 1 \\ e_1 & e_5 & e_6 & e_2 & e_3 & e_4 \end{bmatrix}$$

余树边 & 树枝边

割集矩阵(9)

- 定理3.4.8
  - 设S_f和C_f分别是连通图G中关于某棵树T的基本割集矩阵和基本回路矩阵，且边次序一致，并设 $S_f = (S_{f_{11}} \quad I), C_f = (I \quad C_{f_{12}})$

    则 $S_{f_{11}} = -C_{f_{12}}^T$
  - 由定理3.4.5 $S_e C_e^T = 0$ 易证
- 推论　　已知 $C_{f_{12}} = -B_{11}^T B_{12}^{-T}$
  - 当连通图G的基本割集矩阵与基本关联矩阵的边次序一致时，有 $S_{f_{11}} = B_{12}^{-1} B_{11}$

第三章 树

- 树的有关定义
- 基本关联矩阵及其性质
- 支撑树的计数
- 回路矩阵与割集矩阵
- Huffman树
- 最短树

Huffman树

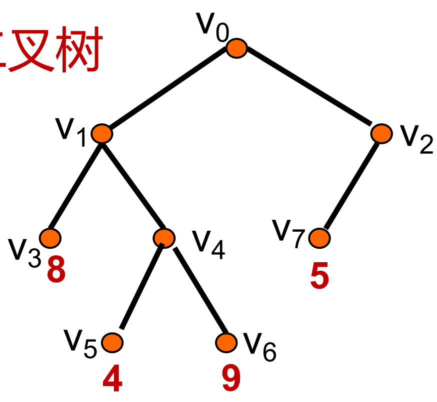
- 例3.6.1
  - 已知13个英文字母构成字符串adacatedecade。试用二进制字符串代替字母，并保证该英文字符串与二进制串能够一一对应。
  - 例如ASCII码
    - a=0x61，c=0x63，……，t=0x74
    - 总长度：8bit * 13=104 bit
  - 其他方法
    - 用0表示a，用1表示d，用00表示c ……？
    - 接收到：010000…

接收端
不能恢复

  - 如何编码(保证能唯一解码)，能够使总长最小？

Huffman树(2)

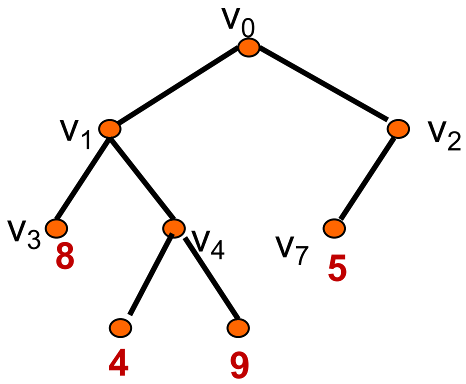
- 定义3.6.1（二叉树与完全二叉树）
  - 除树叶外，其余结点的正度最多为2的外向树称为二叉树
  - 正度都是2的称为完全二叉树
  - 为什么学二叉树而不是三叉树？

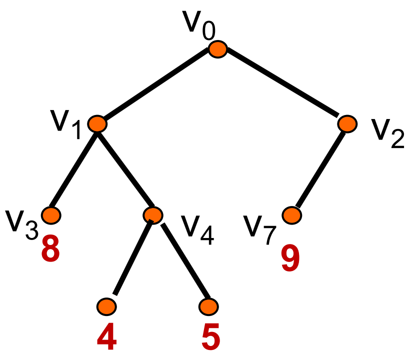


- 赋权二叉树
  - 二叉树T的每一个叶结点v_i都分别赋以一个正实数w_i
- 带权路径总长度 (WPL)
  - 树根v_0到叶结点v_i的路径$P(v_0,v_i)$所包含的边数记为路径的长度l_i，则二叉树T带权的路径长度总长是 $WPL=\sum_i l_i w_i$, v_i是树叶

Huffman树(3)

- 最优二叉树
  - 若给定了树叶数目以及权值，可构造出不同的赋权二叉树
  - 在这些二叉树中，带权路径总长WPL最小的二叉树称为最优二叉树

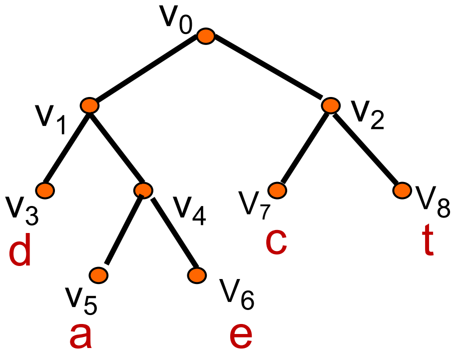


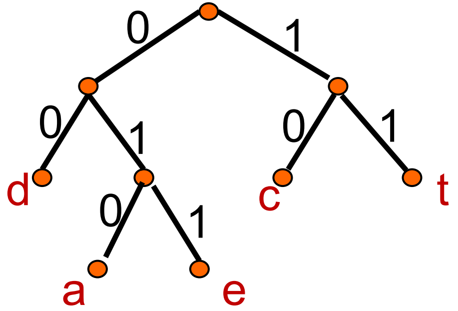


Huffman树(4)

- 例3.6.1
  - 已知英文字符串adacatedecade
  - 试用二进制字符串代替某个字母，并保证该英文字符串与二进制串构成一一对应
  - 解：该字符串中有字母a,d,e,c,t
  - 令每个字母对应二叉树的一个树叶
  - 根到树叶的路径是唯一的，而且这条路径绝不会是树根到另一个树叶路径的一部分
  - 构造一一映射：如令向左的边为0，向右的边为1，则路径与二进制串构成了一一映射

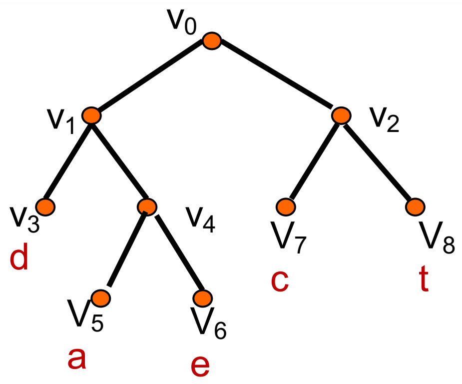
    0100001010010101100011100100001

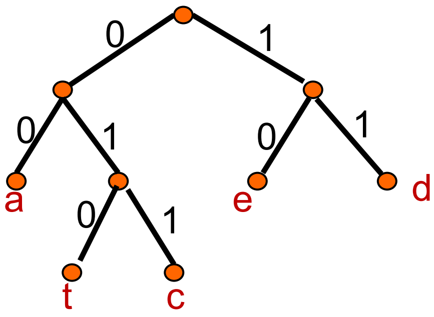




Huffman树(5)

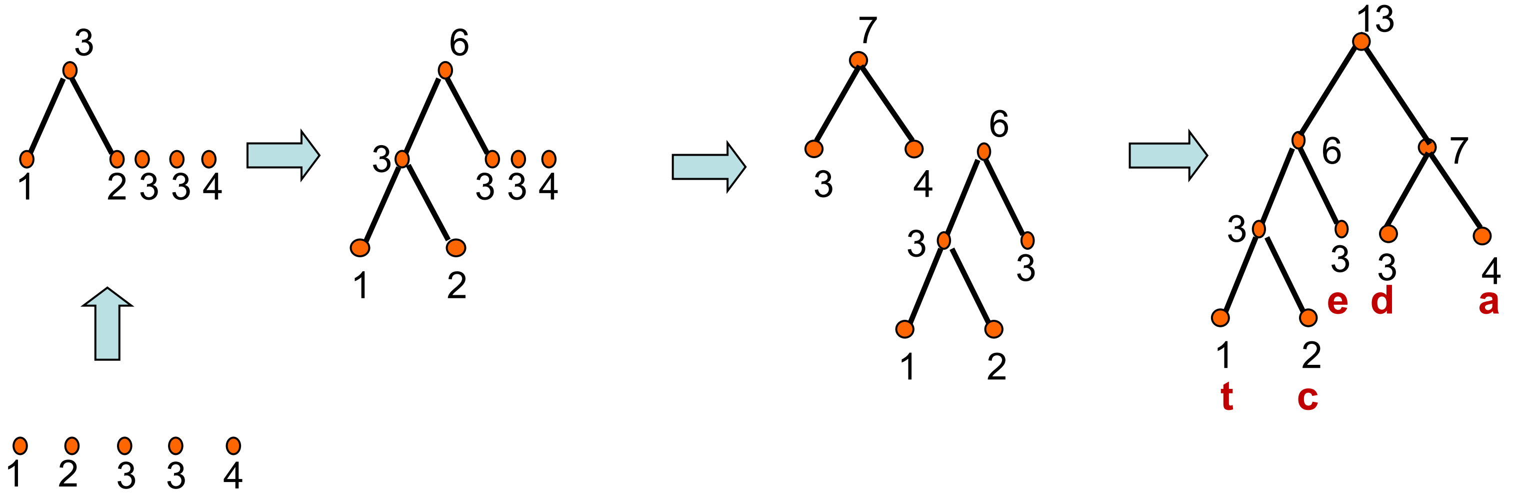
- 例（续）
  - 英文字符串adacatedecade
  - 例如令d, a, e, c, t分别对应左图的v_3, v_5, v_6, v_7, v_8, 则 d←00,a ←010, e ←011,c ←10,t ←11
  - 该英文字符串对应 010000101001010110001110010000011
  - 如果字母与树叶的对应情况如下图，即a←00, t←010, c←011, e ←10, d←11
  - 则对应字符串是 00110001100010101110011001110
  - 这两种情况下字符串的总长分别是33和29
  - 如何构造总长最短的二叉树？
  - 字母a,d,e,c,t分别出现4,3,3,2,1次





Huffman树(6)

- 例3.6.2
  - 字母a,d,e,c,t分别出现4,3,3,2,1次
  - 构造权序列为(1,2,3,3,4)的Huffman树



Huffman树(7)

- Huffman树
  - Huffman给出了构造n个树叶的最优二叉树算法，由此算法得到的树称为Huffman树
- 算法描述如下
  1. 对n个权值(叶子节点)由小到大进行排序，满足
     $w_{i1} \le w_{i2} \le w_{i3} \le \ldots \le w_{in}$
  2. 计算虚拟权值$w_i = w_{i1} + w_{i2}$，作为新增中间结点v_i的权，v_i的左儿子为v_{i1}，右儿子为v_{i2}
  3. 在权序列中删除 w_{i1}, w_{i2} ,按序插入新的虚拟节点w_i，$n=n-1$。当n＝1时结束，否则，转（1）

Huffman树(8)

- 复杂度分析
  - 算法的计算复杂度主要取决于步骤1（n个权值的第一次排序），它一般需进行nlogn次比较
  - 之后每当产生w_i时，只需在新序列中进行插入运算，其复杂度是logn
  - 总共进行n-2次循环
  - 因此整个算法的计算复杂度是O(nlogn)
- 定理3.6.1
  - 由Huffman算法得到的二叉树为最优二叉树

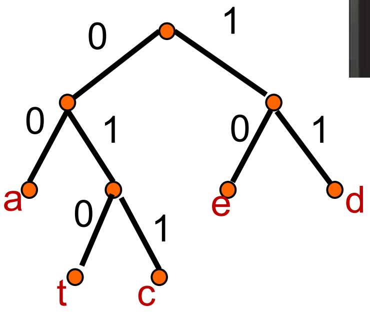
Huffman树(9)

- Huffman树广泛应用于各种压缩算法中
  - ZIP（无损压缩）
  - MP3，JPEG（有损压缩）
- 例：JPEG文件格式

SOI 文件头	……	DHT 定义Huffman表	APP0 JFIF应用数据块	……	EOI 文件尾

- IP分组压缩，流量压缩仪
- 讨论：能唯一解码，是否能抗差错？

0100001010010101100011100100011



视频编码：I帧的利与弊

单选题 1分

设置

字符串“alibaba”的二进制哈弗曼编码有___位（bit）

A 11

B 12

C 13

D 14

提交

第三章 树

- 树的有关定义
- 基本关联矩阵及其性质
- 支撑树的计数
- 回路矩阵与割集矩阵
- Huffman树
- 最短树

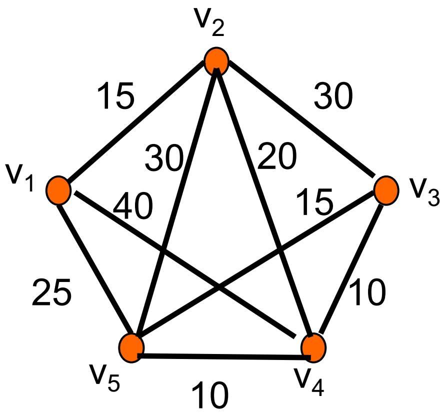
最短树

- 铺设输油管道
  - 已知任两个加油站之间输油管道的铺设费用，要让每个站都能供应上油，怎么铺？
- 最短树和最长树问题
  - 赋权连通图中，总长最小的支撑树叫最短树
  - 在赋权连通图中，求总长最小或最大的支撑树

最短树(2)

- 最短树求解基本思路（Kruskal算法）
  - 不断往边集T中加入全图最短边
  - 如果此时会构成回路，那么它一定是这个回路中的最长边，删之
  - 直至最后达到n-1条边为止
  - 这时T中不包含任何回路，因此构造出最短树。
- 最短路径
  - Dijkstra算法：每次加入到根结点最近的新结点
- 旅行商问题（便宜算法）
  - 不断加入距当前初级回路最近结点的启发式算法

“AI”



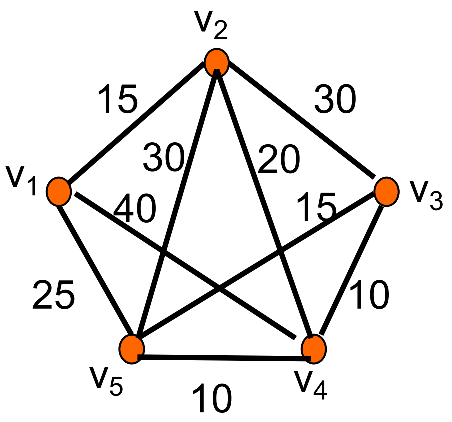
最短树(3)

- Kruskal算法
  1. T←Φ
  2. 当|T|<n-1且E≠Φ时，
  3. Begin （迭代）
  4. e ← E中最短边
  5. E ← E-e
  6. 若T+e无回路，则T ← T+e
  7. end
  8. 若|T|<n-1，则G非连通，否则输出最短树

算法执行过程中，不需要保证T是连通子图

最短树(4)

- 例3.7.1
  - 执行Kruskal算法的过程
    - T←(v_4,v_3), (v_4,v_5), (v_1,v_2)
    - 当加入(v_3,v_5)时构成回路，因此边(v_3,v_5)不加入T
    - 此后T ←T+ (v_2,v_4)
    - 这时|T|=n-1,T={(v_4,v_3), (v_4,v_5),(v_1,v_2), (v_2,v_4)}, 结束

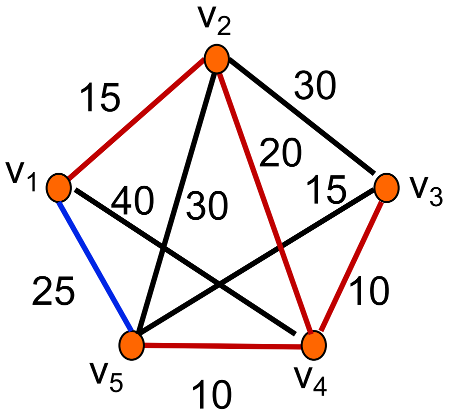


最短树(5)

- Kruskal算法计算复杂度
  - Kruskal算法的计算复杂性主要取决于步骤4和6
  - 对m条边的权采用堆结构存放，可以保证根结点是当前的最小权。
    - 堆结构是一种均衡二叉树，它满足对于任何一个父亲结点，其权都小于其左右儿子的权。
    - 建堆的计算复杂性是O(m)
  - 步骤4找最短边的计算复杂性是O(logm)
  - 步骤2-6迭代次数p
- 定理3.7.2
  - Kruskal算法计算复杂性是O(m+plogm)

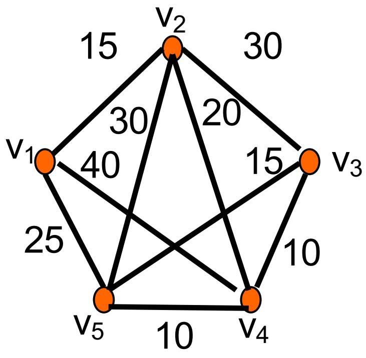
最短树(6)

- 定理3.7.1
  - T=(V,E')是赋权连通图G=(V,E)的最短树，当且仅当对任意的余树边e∈E-E'，满足其边权w(e) ≥w(a),其中a为对应回路 $C^e(C^e \subseteq E'+e)$ 的任意树枝边a∈C^e (a≠e)
- 证明
  - 必要性: (反证法)
    - 如果存在一条余树边e,满足w(e)<w(a),a∈C^e
    - 则T⊕{a,e}得到新树T'比T更短，与T是最短树矛盾



最短树(8)

- Kruskal算法
  - 不断的往T(非连通子图)中加入当前的最短边e，直到T中包含n-1条无回路的边
  - T是最短边的集合（构造过程不保证连通性）



- Prim算法
  - 首先初始化集合V'为任选一结点v_0
  - 然后不断在V-V'中选一条距离V'中任意点(如点v)最近的节点u进入树T (连通子树)，并令V'=V'+u, 直至V'=V

最短树(9)

- 算法描述(初选v_1)：
  1. $t \leftarrow v_1$, $T \leftarrow \Phi$,$U \leftarrow \{t\}$ //T为部分建成的连通子树，U为T的结点集，t为代表当前T的虚拟结点
  2. while U≠V do // V为原图结点集
     begin
  3. $W(t,u) = \min_{v \in V-U}\{w(t,v)\}$ //找: 到T的最近结点u
  4. $T \leftarrow T+e(t,u)$
  5. $U \leftarrow U+u$
  6. for $v \in V-U$ do
  7. $w(t,v) \leftarrow \min\{w(t,v), w(u,v)\}$ //更新剩余各结点v到T的最短距离
     end

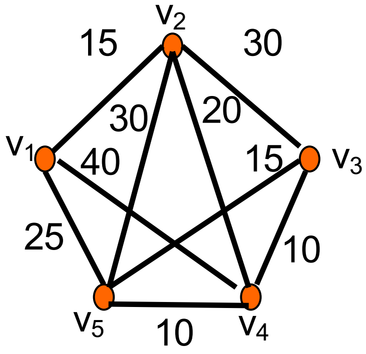
每次查找连接两个点集的最短边

不是到树根的最短距离！

Prim算法的计算复杂度是多少？ **O(n²)**

最短树(10)

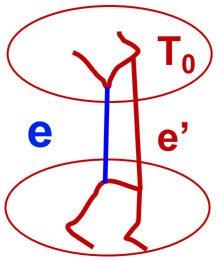
- 例3.24(设首选$U=\{v_1\}$)
  - 3.$\min\{w(v_1,v_i)\}=w(v_1,v_2)=15, U=\{v_1\}+v_2$
  - 6.$w(t,v_3)=w(v_2,v_3)=30$, $w(t,v_4)=w(v_2,v_4)=20$, $w(t,v_5)=w(v_1,v_5)=25$
  - 3.$\min\{w(t,v_i)\}=w(v_2,v_4)=20, U=\{v_1,v_2\}+v_4$
  - 6.$w(t,v_3)=w(v_4,v_3)=10$, $w(t,v_5)=w(v_4,v_5)=10$
  - 3. $\min\{w(t,v_i)\}=w(v_4,v_3)=10, U=\{v_1,v_2,v_4\}+v_3$
  - 6. $w(t,v_5)=w(v_4,v_5)=10$
  - 3. $\min\{w(t,v_i)\}=w(v_4,v_5)=10, U=\{v_1,v_2,v_4,v_3\}+v_5=V$
  - 结束
  - 因此最短树$T=\{ (v_1,v_2) , (v_2,v_4) ,(v_4,v_3), (v_4,v_5) \}$



每次加入
两个点集间的最短边

最短树(11)

- 定理3.7.3
  - 设V'是赋权连通图G=(V,E)的结点真子集，e是两端点分跨在V'和V-V'的最短边，则G中一定存在包含e的最短树T
  - 证明(构造法，注意最短树不唯一)：
    - 设T_0 是G的一棵最短树，若上述e不属于T_0，则T_0+e构成唯一回路
    - 该回路一定包含e和至少一条分跨在V'和V-V'的边e'=(u,v)
    - 由已知条件w(e)≤ w(e')，作$T_0 \oplus \{e,e'\}$，得到的仍然是最短树
    - 因此G中存在包含e的最短树T



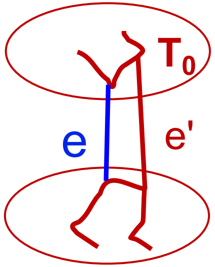
树变换

最短树(12)

- 定理3.7.4
  - Prim算法的结果是赋权连通图G的一棵最短树
- 证明：
  - 首先证明是一棵支撑树（n-1条边和n个节点）
    - 采用归纳法，初始$U=\{v_1\}$，T= Φ，它是由U导出的树
    - 设|U|=i，T是U导出的树
    - 则下一次迭代时，U中增加一新结点u，T中也加入一条与u 相连的边
    - 因此T连通，有|U|-1条边，它是由U导出的一棵树
    - 因此最终T是G的支撑树

最短树(13)

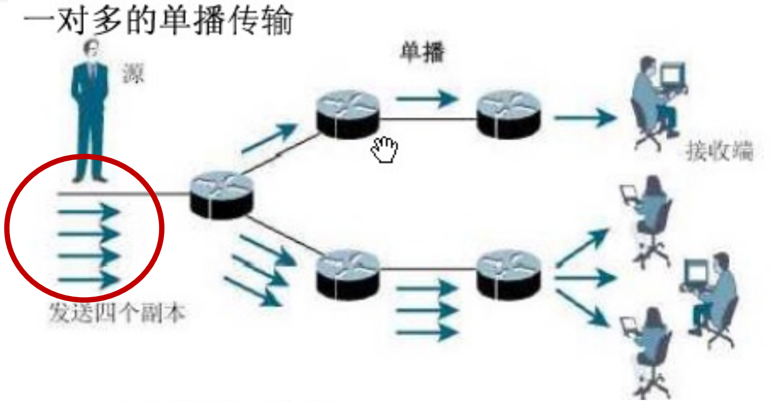
- 证（续）
  - 再证明Prim算法产生的树T是一棵最短树
    - 设T_0 是G的一棵最短树
    - 若$T \neq T_0$，将T_0变换为Prim算法产生的T
    - 对任意的$e \in T-T_0$，Prim算法加入的每条边e都是一条连接两个节点集的最短边
    - 由定理3.7.3，对任意的 $e \in T-T_0$ ，一定能构造最短树 $T'=T_0 \oplus (e,e')$，其中$e' \in C^e \cap T_0$
    - 继续对T'如此处理，直至最终T'=T，它仍然是最短树

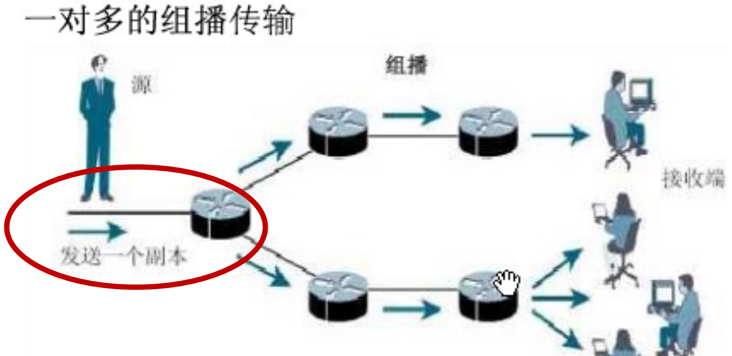


最短树(14)

- 最短树算法
  - Kruskal算法复杂性O(m+plogm)，与迭代次数p和边数m相关
  - Prim算法复杂性$O(n^2)$仅与节点数相关
- 两个算法的适用范围
  - 稀疏图/边少（Kruskal）v.s. 稠密图/点少（Prim）
- 最长树问题
  - 构造新图：给定大数减边权作为新权值
  - Kruskal算法(原为加入当前的最短边)：将加入树的边次序按边权构成非增序列
  - 最短路径Dijkstra算法->最长路径：边数不定，不适用大数减

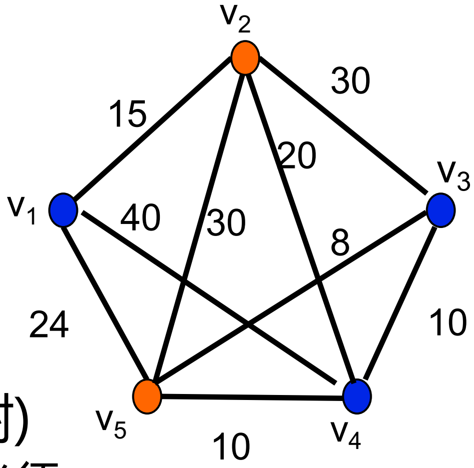
最短树(15)





互联网组播传输模式

- 互联网基本传输模式
  - 单播：Unicast
  - 广播：Broadcast
  - 组播：Multicast
- D算法
  - 单播：最短路径树
- 最优组播树(Steiner树)
  - 不必包含所有节点而必须包含组播组成员的最短树
  - NPC问题
  - 可否多项式时间求解？
    启发式算法，满足三角不等式、节点度约束



全集 v.s. 子集

总结：第三章 树

- 树的有关定义
- 基本关联矩阵及其性质
- 支撑树的计数
- 回路矩阵与割集矩阵
- 应用于编码的最优树
  - Huffman树：构造最小的带权路径总长度
- 最短树
  - 从AI开始两个思路：Kruskal和Prim算法

作业

- P85 习题三
  1. 回路矩阵：基本回路矩阵27(1)，基本割集矩阵27(2)
  2. Huffman树：34
  3. Huffman树：为什么学二叉树，而不是三叉树或一叉树？
  4. 最短树：39
  5. 如果我是欧拉
     - 列表总结割集矩阵的创新思路（并请与回路矩阵对比）
     - 可以包括但不限于概念、代数表示、定义、定理等部分