

# 关键路径

吉林大学计算机学院  
谷方明

fmgu2002@sina.com



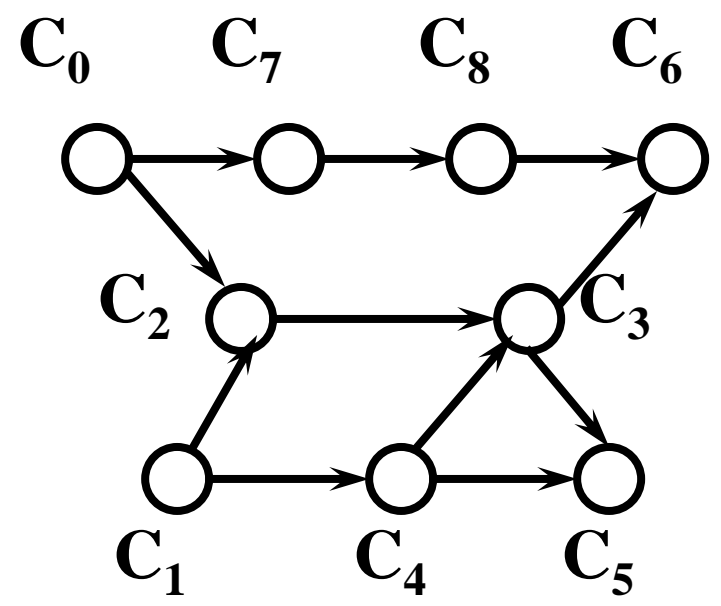


# 学习目标

- 掌握**AOE**网、关键路径等概念的定义。
- 掌握关键路径的求解方法、技巧及相关分析。

# 计算机专业必修课程

课号	课名	先修	学期数
C <sub>0</sub>	高等数学	无	3
C <sub>1</sub>	程序设计基础	无	1
C <sub>2</sub>	离散数学	C <sub>0</sub> ,C <sub>1</sub>	2
C <sub>3</sub>	数据结构	C <sub>2</sub> ,C <sub>4</sub>	1
C <sub>4</sub>	程序设计语言	C <sub>1</sub>	1
C <sub>5</sub>	编译技术	C <sub>3</sub> ,C <sub>4</sub>	2
C <sub>6</sub>	操作系统	C <sub>3</sub> ,C <sub>8</sub>	2
C <sub>7</sub>	普通物理	C <sub>0</sub>	2
C <sub>8</sub>	计算机原理	C <sub>7</sub>	2



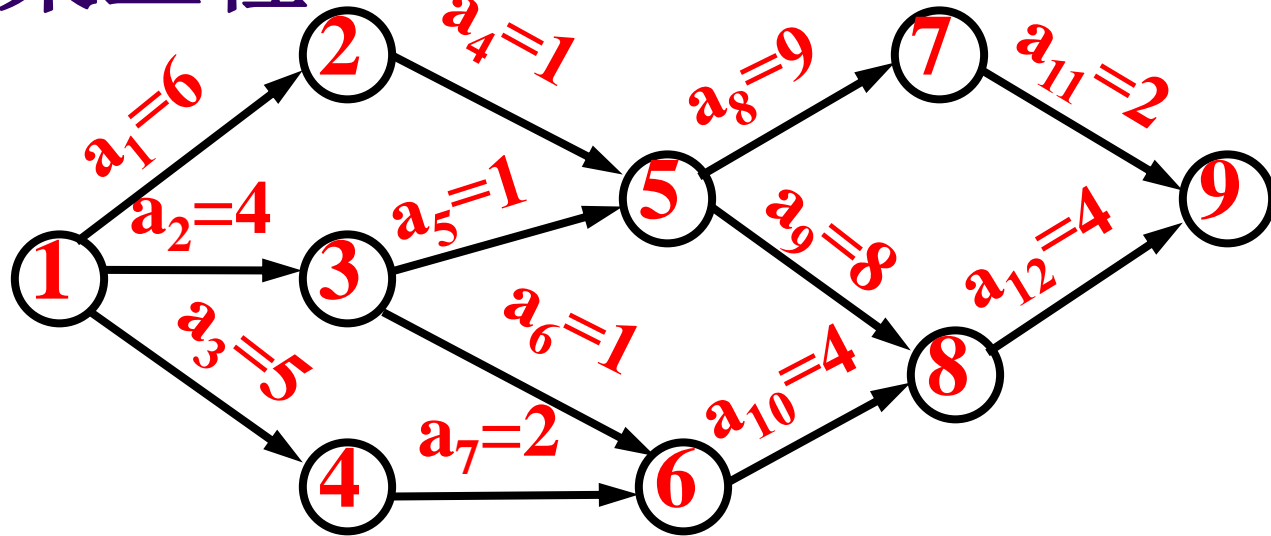


# AOE网

- 时间约束
- 边表示活动(**Activity**)
- 边的权值表示活动的持续时间(**Duration**)
- 顶点表示入边的活动已完成，出边的活动可以开始的状态，也称为事件(**Event**)
- 这样的有向无环带权图叫做**AOE (Activity On Edges)**网。

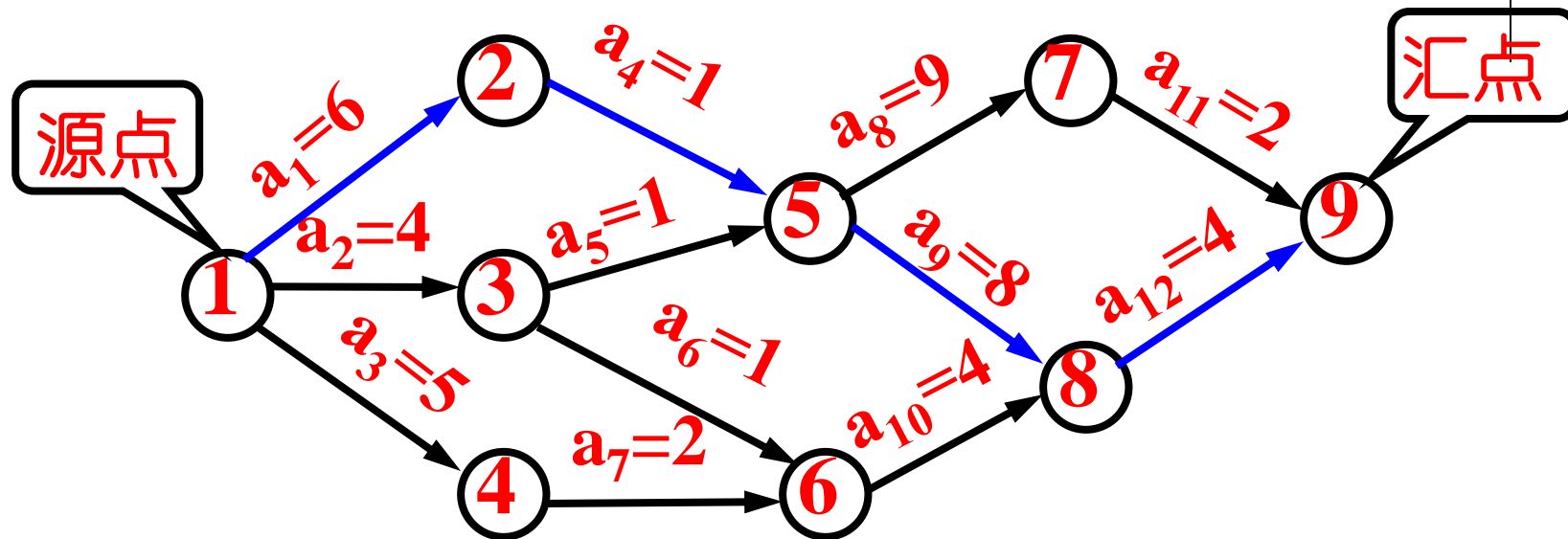


## [例] 某工程



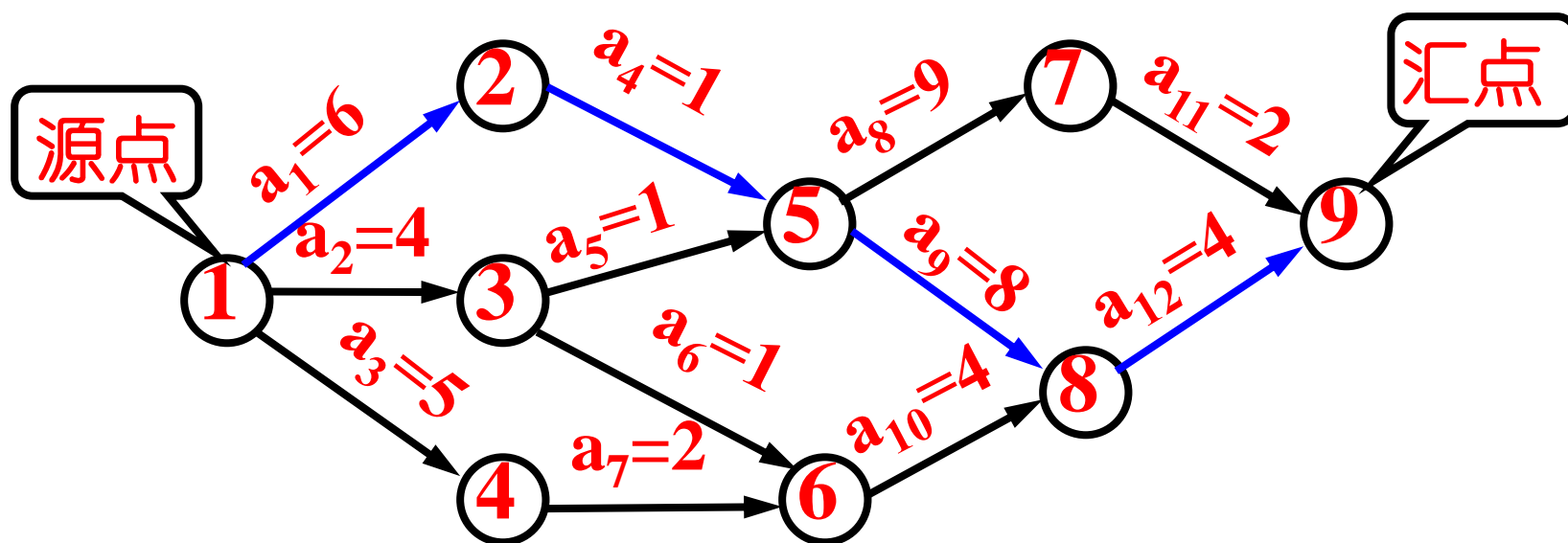
- **AOE网中**，有些活动可并行进行。只有各条路径上所有活动都完成了，整个工程才算完成。
- 完成整个工程至少需要多少时间？
- 为缩短工程的时间，应当加快哪些活动？
- 为了不延误整个工期，哪些活动不得延期，哪些可适当延期

# 关键路径和关键活动

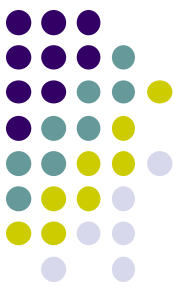


- **源点**: 表示整个工程的开始（入度为零）
- **汇点**: 表示整个工程的结束（出度为零）
- 完成整个工程所需的时间取决于从源点到汇点的最长路径长度。

- 关键路径： AOE网中具有最大长度的路径。
- 关键活动： 不得延期的活动。不按期完成就会影响整个工期的活动。



- 关键活动： 关键路径上的活动；
- 关键路径： 从源点到汇点由关键活动构成的路径。



# 关键路径有关的量

①从源点 $v_0$ 到 $v_i$ 的最长路径长度  
为事件 $v_i$ 的最早发生时间  $ve(i)$

$$ve(1)=0$$

$$ve(2)=6$$

$$ve(3)=4$$

$$ve(4)=5$$

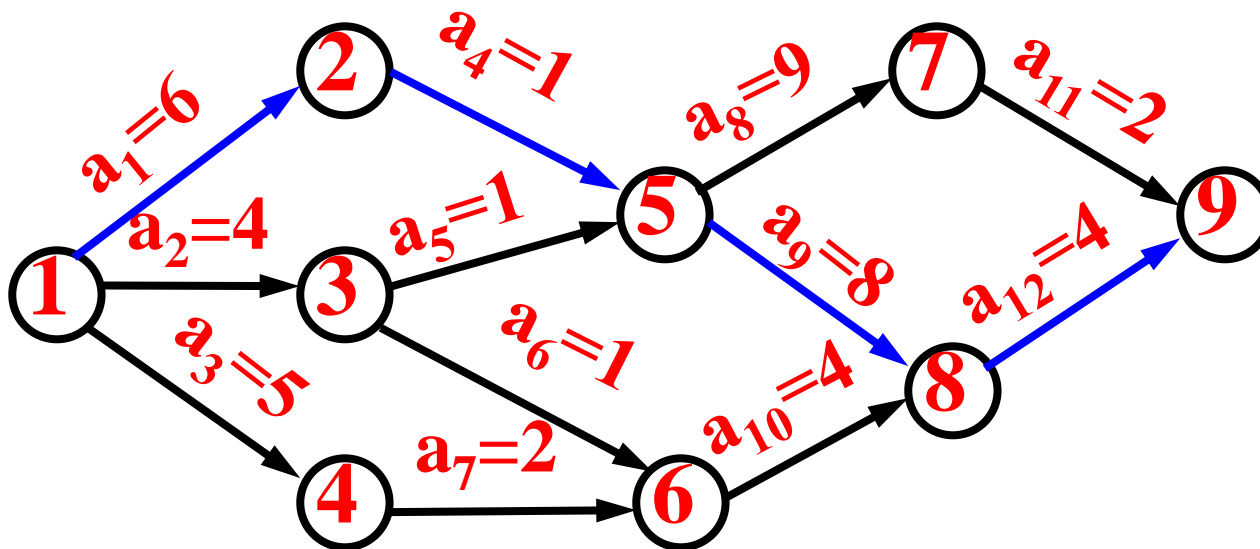
$$ve(5)=7$$

$$ve(6)=7$$

$$ve(7)=16$$

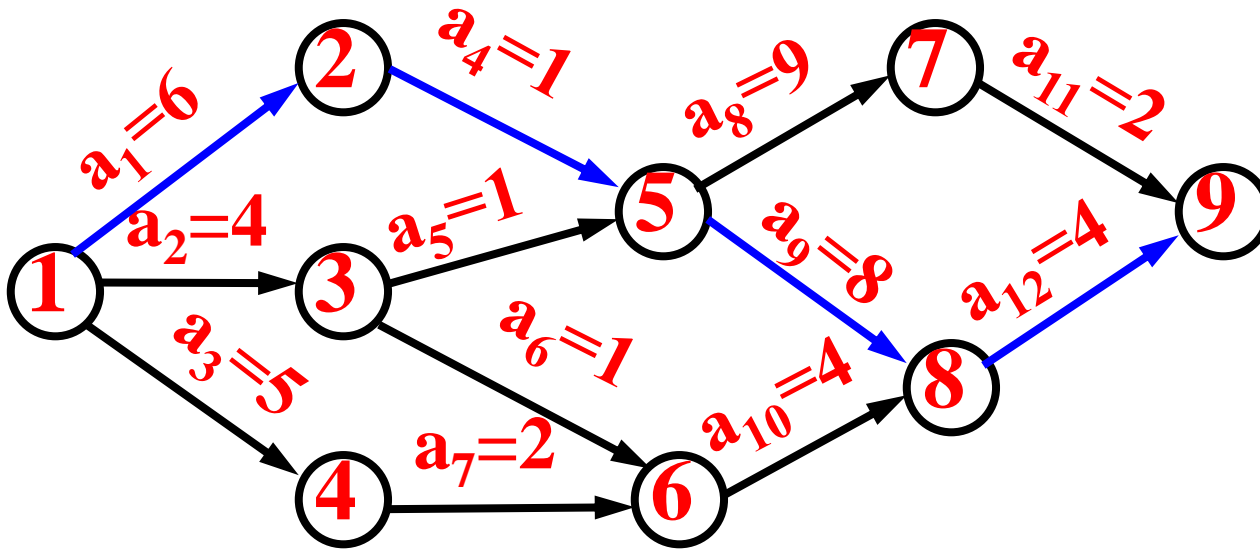
$$ve(8)=15$$

$$ve(9)=19$$



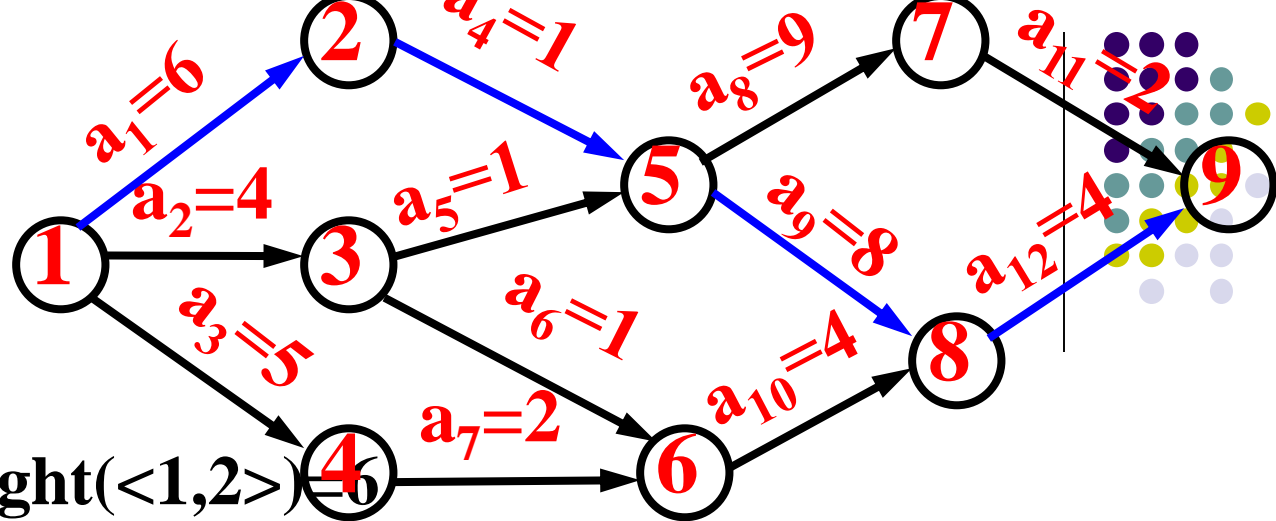


# [例] 求关键路径 — 第1步 $ve(i)$



按拓扑序递推

$$ve(k) \begin{cases} ve(1) = 0 & k=1 \\ \max\{ve(j) + \text{weight}(\langle j, k \rangle)\} \\ \langle v_j, v_k \rangle \in E(G), k=2, 3, \dots, n \end{cases}$$



$$ve(1)=0$$

$$ve(2)= ve(1)+weight(<1,2>)=6$$

$$ve(3)= ve(1)+weight(<1,3>)=4$$

$$ve(4)= ve(1)+weight(<1,4>)=5$$

$$ve(5)= \max\{ve(2)+ weight(<2,5>), \\ ve(3)+ weight(<3,5>)\}=\max\{6+1,4+1\}=7$$

$$ve(6)= \max\{ve(3)+ weight(<3,6>), \\ ve(4)+ weight(<4,6>)\}=\max\{4+1,5+3\}=7$$

$$ve(7)= ve(5)+weight(<5,7>)=7+9=16$$

$$ve(8)= \max\{ve(5)+ weight(<5,8>), \\ ve(6)+ weight(<6,8>)\}=\max\{7+8,7+4\}=15$$

$$ve(9)= \max\{ve(7)+ weight(<7,9>), \\ ve(8)+ weight(<8,9>)\}=\max\{16+2,15+4\}=19$$

# 关键活动有关的量

② 事件 $v_j$ 的最迟发生时间  $vl(j)$ :

保证汇点的最早发生时间不推迟的前提下，事件 $v_j$ 允许的最迟开始时间，等于 $ve(n)$ 减去从 $v_j$ 到 $v_n$ 最长路径长度。

$$vl(9) = 19$$

$$vl(8) = 15$$

$$vl(7) = 17$$

$$vl(6) = 11$$

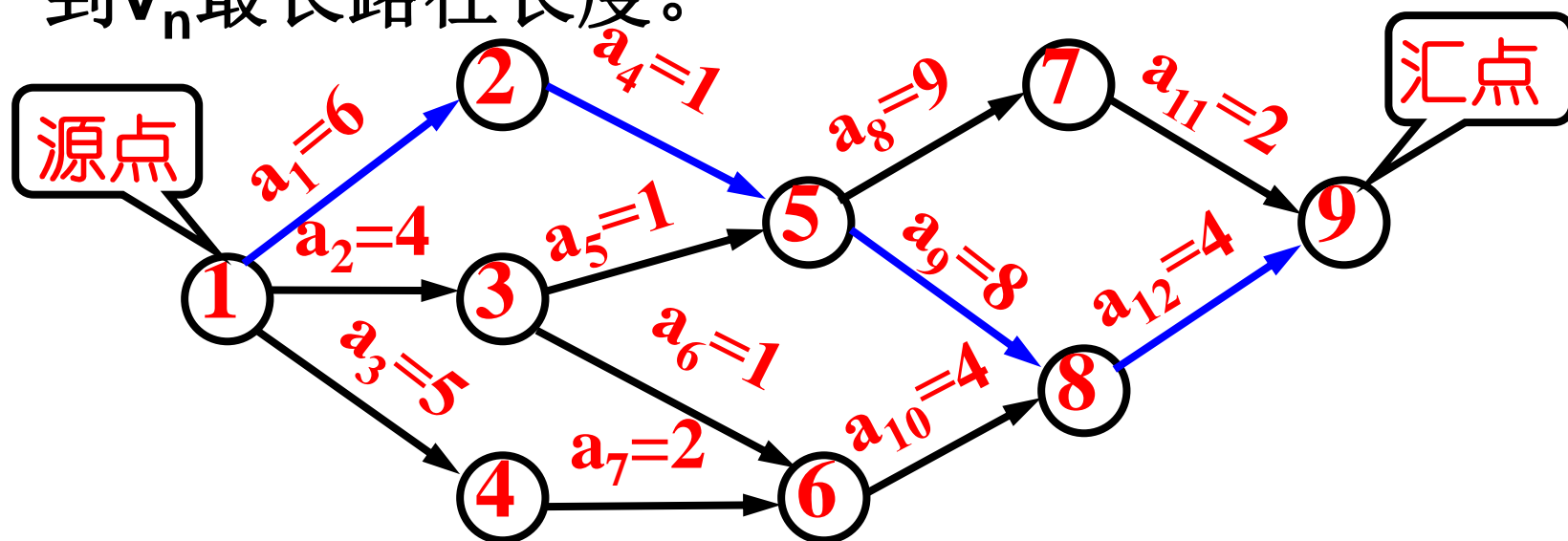
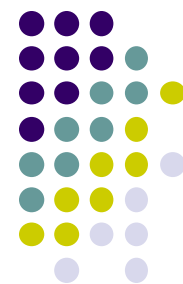
$$vl(5) = 7$$

$$vl(4) = 9$$

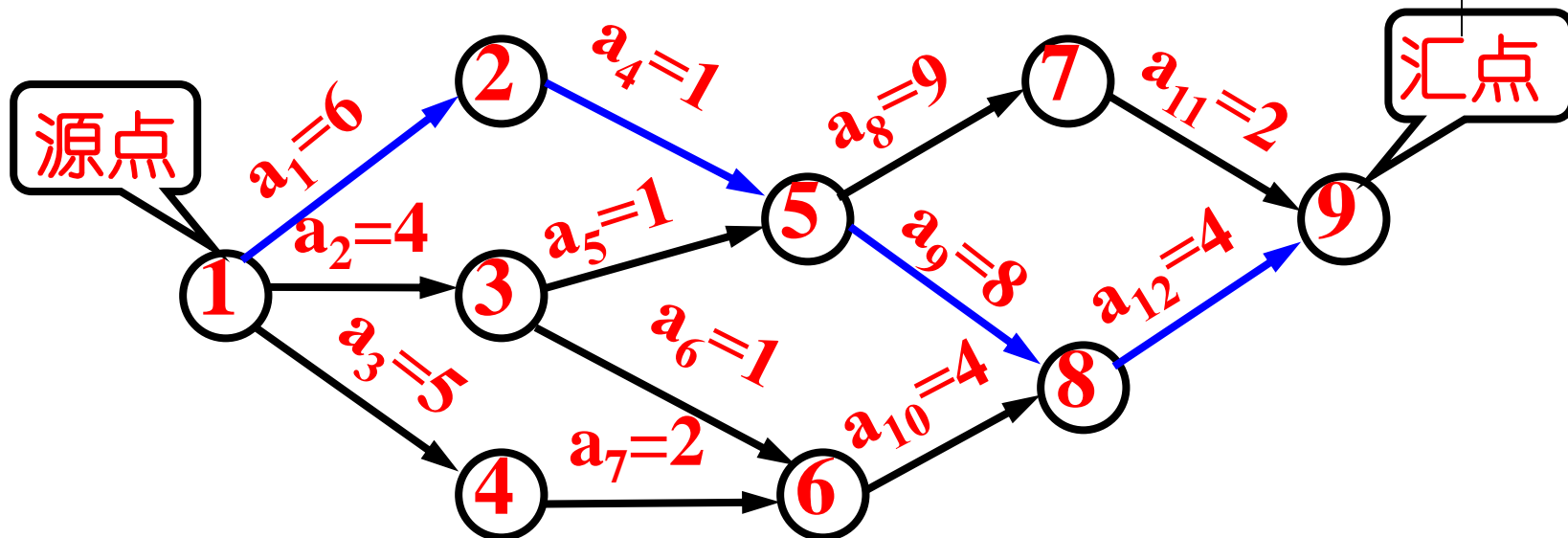
$$vl(3) = 6$$

$$vl(2) = 6$$

$$vl(1) = 0$$

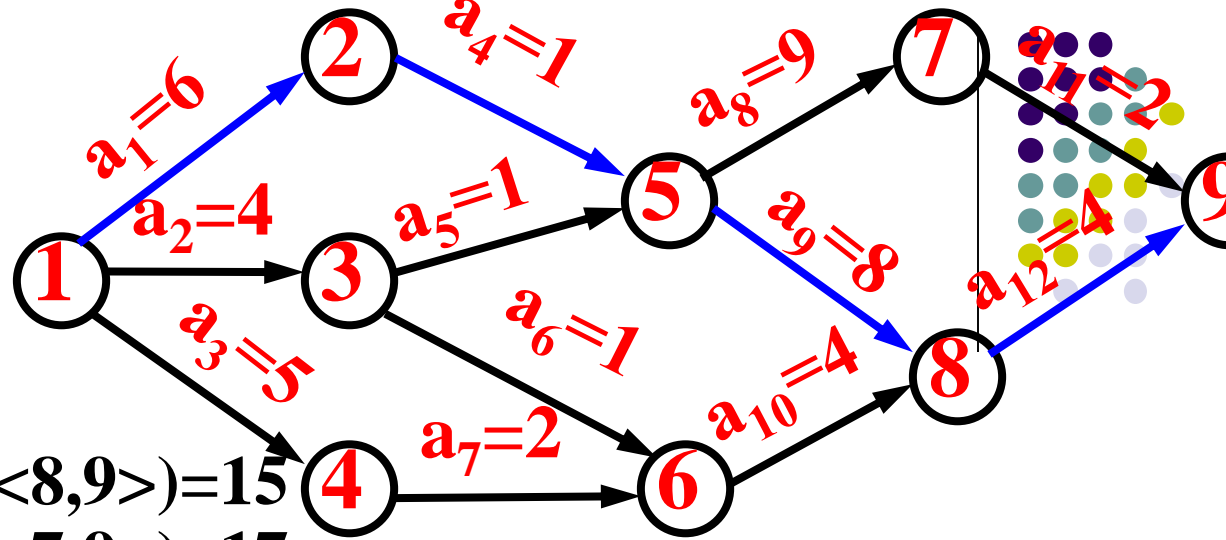


## [例] 求关键路径 — 第2步 $vl(k)$



按逆拓扑序递推：

$$vl(j) \begin{cases} ve(n) & j=n \\ \min\{vl(k) - \text{weight}(\langle j, k \rangle)\} \\ & \langle v_j, v_k \rangle \in E(G), j = n-1, n-2, \dots, 1 \end{cases}$$



$$vl(9) = ve(9) = 19$$

$$vl(8) = vl(9) - \text{weight}(<8,9>) = 15$$

$$vl(7) = vl(9) - \text{weight}(<7,9>) = 17$$

$$vl(6) = vl(8) - \text{weight}(<6,8>) = 11$$

$$vl(5) = \min\{vl(8) - \text{weight}(<5,8>), \\ vl(7) - \text{weight}(<5,7>)\} = \min\{15 - 8, 16 - 9\} = 7$$

$$vl(4) = vl(6) - \text{weight}(<4,6>) = 11 - 2 = 9$$

$$vl(3) = \min\{vl(6) - \text{weight}(<3,6>), \\ vl(5) - \text{weight}(<3,5>)\} = \min\{11 - 1, 7 - 1\} = 6$$

$$vl(2) = vl(5) - \text{weight}(<2,5>) = 7 - 1 = 6$$

$$vl(1) = \min\{vl(2) - \text{weight}(<1,2>), \\ vl(3) - \text{weight}(<1,3>), \\ vl(4) - \text{weight}(<1,4>)\} = \min\{6 - 6, 6 - 4, 9 - 5\} = 0$$



# 关键路径有关的量

③ 活动 $a_i$ 的最早开始时间 $e(i)$ :

设活动  $a_i$  为有向边 $\langle v_j, v_k \rangle$ , 则  $e(i) = ve(j)$ 。

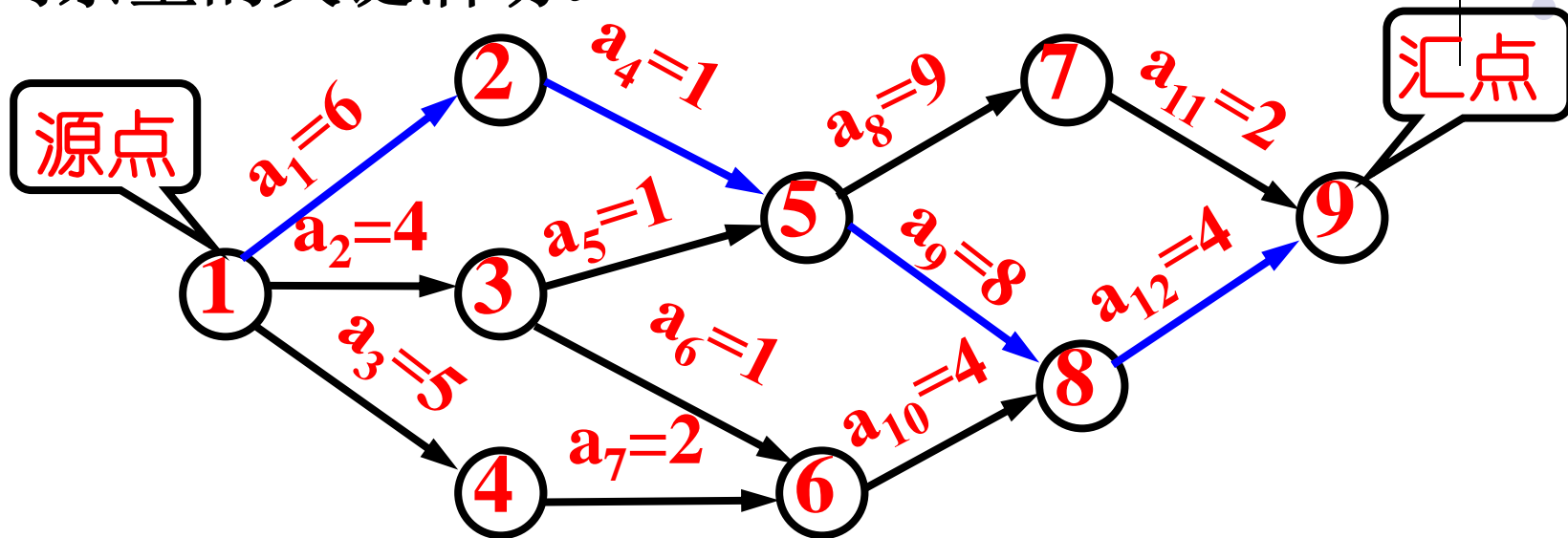
✓  $ve(j)$  是从源点 $v_0$ 到 $v_j$ 的最长路径长度, 决定了所有从 $v_j$ 开始的活动的最早开始时间。

④ 活动 $a_i$ 的最迟开始时间  $l(i)$ :

设活动  $a_i$  为有向边 $\langle v_j, v_k \rangle$ , 则  $l(i) = vl(k) - \text{weight}(\langle j, k \rangle)$ 。

✓  $l(i)$  是在不会引起工期延误的前提下, 该活动允许的最迟开始时间。

关键活动:  $l(i) = e(i)$  表示活动 $a_k$  是没有时间余量的关键活动。

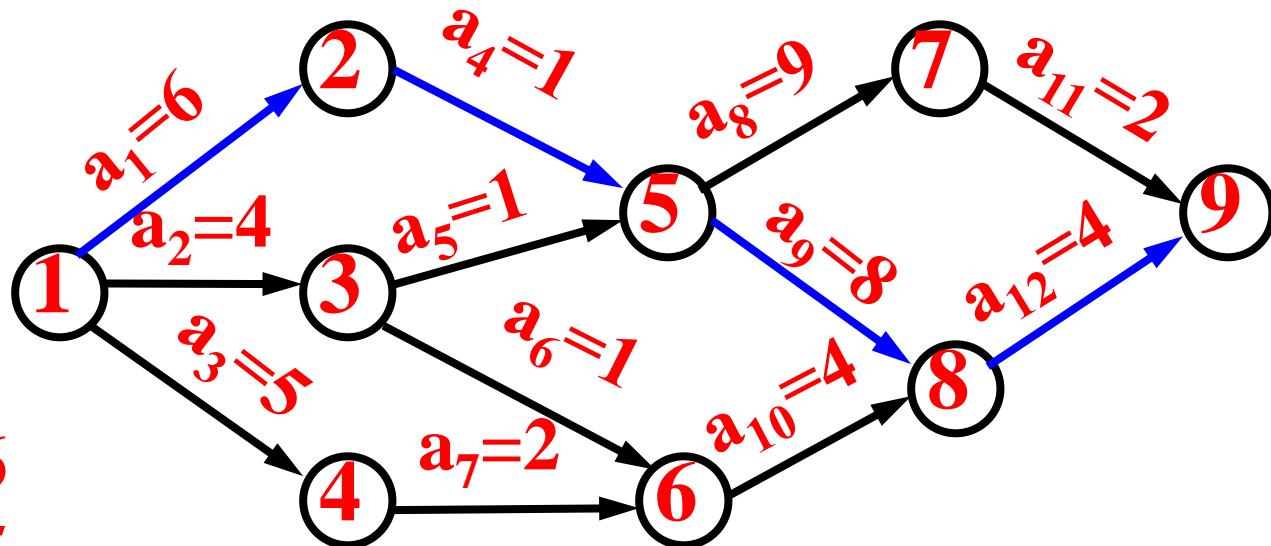


为找出关键活动, 需要求各个活动的  $e(i)$  与  $l(i)$ , 以判别是否  $l(i) = e(i)$

为求得  $e(i)$  与  $l(i)$ , 需要先求得从源点  $V_0$  到各个顶点  $V_j$  的  $ve(j)$  和  $vl(j)$ 。

# [例] 求关键路径 — 第3步

$ve(1)=0$   
 $ve(2)=6$   
 $ve(3)=4$   
 $ve(4)=5$   
 $ve(5)=7$   
 $ve(6)=7$   
 $ve(7)=16$   
 $ve(8)=15$   
 $ve(9)=19$



$vl(9)=19$   
 $vl(8)=15$   
 $vl(7)=17$   
 $vl(6)=11$   
 $vl(5)=7$   
 $vl(4)=9$   
 $vl(3)=6$   
 $vl(2)=6$   
 $vl(1)=0$

$$e(i)=ve(j), \quad l(i)=vl(k)-weight(<j,k>)$$

$a_i$	$a_1$	$a_2$	$a_3$	$a_4$	$a_5$	$a_6$	$a_7$	$a_8$	$a_9$	$a_{10}$	$a_{11}$	$a_{12}$
$e(i)$	0	0	0	6	4	4	5	7	7	7	16	15
$l(i)$	0	2	4	6	6	10	9	8	7	11	17	15
$l_i-e_i$	0	2	2	0	2	5	3	1	0	4	0	0





# 求关键活动算法

- ① 对**AOE**网拓扑排序; (若网中有回路, 则终止算法)
- ② 按**拓扑次序**求出各顶点事件的最早发生时间**ve**;
- ③ 按拓扑序列的逆序求各顶点事件的最迟发生时间**vl**;
- ④ 根据**ve**和**vl**的值, 求各活动的最早开始时间**e(i)**与最迟开始时间**l(i)**, 若**e(i)=l(i)**, 则**i**是关键活动。



# 图的关键路径算法

算法CriticalPath ( )

/\* 关键路径核心算法,假定源点1、汇点n、拓扑序1~n \*/

**CPath2**[计算事件的最早发生时间]

for ( i = 1 ; i <= n ; i ++ ) ve [i] = 0;

for ( i = 1 ; i <= n-1 ; i ++ ) /\*按拓扑序递推求ve[i]\*/

for ( p = Head [i] . adjacent ; p ; p = p-> link) {

k = p -> VerAdj ;

if (ve[i] + p -> cost > ve[k] )

ve[k] =ve[i] + p -> cost ;

} //不必存逆图；前点处理时更新后点，下标1..n-1



### **CPath3**[计算事件的最迟发生时间]

```
for ( i = 1 ; i <= n ; i ++ ) vl [i] = ve [n];
```

```
for ( i = n-1 ; i >= 1 ; i -- ) /*按逆拓扑序求vl*/
```

```
    for ( p = Head [i] . adjacent ; p ; p = p-> link) {
```

```
        k = p -> VerAdj ;
```

```
        if ( vl[k] - p-> cost < vl[i] )
```

```
            vl[i] = vl[k] - p-> cost ;
```



**CPath4**[求诸活动的最早开始时间和最迟开始时间]

```
for ( i = 1 ; i <= n ; i ++ )  
    for ( p = Head [i] . adjacent ; p ; p = p-> link ) {  
        k = p -> VerAdj ;  
        e = ve[i] ;  
        l = vl[k] - cost(p) ;  
        if ( l == e )  
            cout << "< i, k> is Critical Activity! "  
    } ■
```



# 算法分析

## □ 增加拓扑排序

- ✓ 调用算法TopoOrder( )，若返回false，则有环，CPath算法结束。否则，拓扑排序的结果存放在torder[ ]。
- ✓ 更改CPath算法的2、3两步。即在第2、3计算循环中引入变量 j，使j=torder[i];然后用 j 代替 i 进行计算。

## □ 时间复杂性

- ✓ 进行拓扑排序的时间复杂性为 $O(n+e)$ ,
- ✓ 以拓扑序求ve[i]和以逆拓扑序求vl[i], 所需时间均为 $O(e)$ ,
- ✓ 求各个活动的e[k]和l[k]的时间复杂度为 $O(e)$ , 整个算法的时间复杂性是 $O(n+e)$



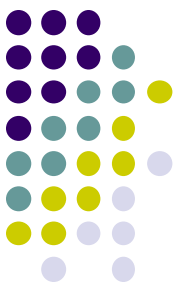
# 算法CriticalPath的说明

## □ CriticalPath正确性

- ✓ 关键路径和关键活动的定义
- ✓  $ve$ 的递推公式。

□ **定理6.3:**任意的非空AOE网至少存在一条关键路径。 **请忽略该定理。**

□ **定理6.3（新）：** 关键路径上的顶点满足  $ve(i) = vl(i)$  。



引理：假设AOE网的事件集为 $\{T_1, T_2, \dots, T_n\}$ ,  $n \geq 1$ . 对任意 $i$ 有 $vl(i) \geq ve(i)$ .

- 证明：由 $vl(i)$ 和 $ve(i)$ 的定义知，对任意出度为零的顶点上式显然成立。假设顶点 $T_i$ 的所有后继顶点也满足上式，根据 $vl$ 的定义知存在顶点 $T_j$ 和边 $\langle T_i, T_j \rangle$ , 且

$$vl(i) = vl(j) - \text{weight}(\langle T_i, T_j \rangle)$$

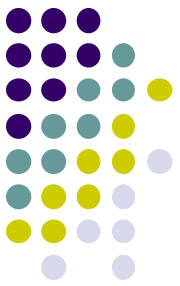
又由 $ve$ 的定义知：

$$ve(j) \geq ve(i) + \text{weight}(\langle T_i, T_j \rangle)$$

$$ve(j) - \text{weight}(\langle T_i, T_j \rangle) \geq ve(i)$$

由归纳假设知 $vl(j) \geq ve(j)$ , 从而

$$\begin{aligned} vl(i) &= vl(j) - \text{weight}(\langle T_i, T_j \rangle) \\ &\geq ve(j) - \text{weight}(\langle T_i, T_j \rangle) \geq ve(i) \end{aligned}$$



## 定理6.3（新）证明

假定路径  $Ti_1, Ti_2, \dots, Ti_k$  是所有从入度为零的顶点到出度为零的顶点的具有最大路径长度的路径。由于该路径具有最大路径长度，因此

$ve(i_1)=0, ve(i_2)=ve(i_1)+weight(< Ti_1, Ti_2 >), \dots,$

$$ve(i_j)=ve(i_{j-1})+weight(< Ti_{j-1}, Ti_j >), \dots \quad (6-2)$$

当  $j=k$  时，显然有  $ve(i_k)=vl(i_k)$

假定当  $j>m$  时有  $ve(i_j)=vl(i_j)$ ，并假定  $Ti_m$  发出的边至少有两条（只有一条，显然得证）。不妨设是  $< Ti_m, Ti_{m+1} >$  和  $< Ti_m, T_y >$ ，如图所示。



由 $ve$ 的定义知

$$ve(y) \geq ve(i_m) + \text{weight}(< Ti_m, T_y >)$$

由引理,得

$$vl(y) - \text{weight}(< Ti_m, T_y >) \geq ve(i_m) \quad (6-3)$$

又由(6-2)式得

$$ve(i_m) = ve(i_{m+1}) - \text{weight}(< Ti_m, Ti_{m+1} >)$$

再由归纳假设有

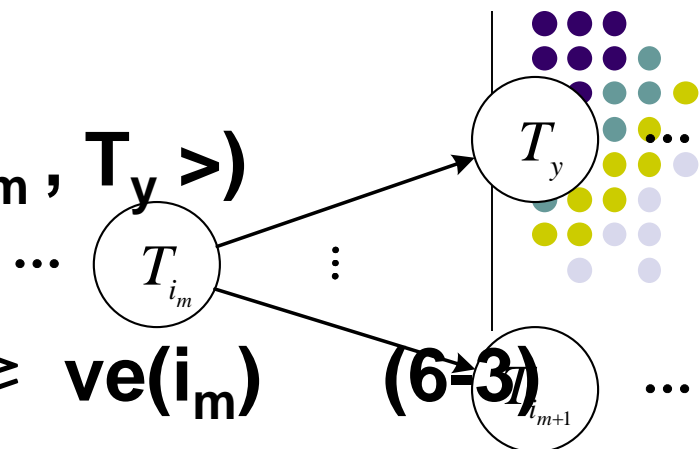
$$ve(i_m) = vl(i_{m+1}) - \text{weight}(< Ti_m, Ti_{m+1} >)$$

从而结合(6-3)式得 :

$$vl(i_{m+1}) - \text{weight}(< Ti_m, Ti_{m+1} >) \leq vl(y) - \text{weight}(< Ti_m, T_y >)$$

由 $y$ 的任意性知:

$$vl(i_m) = vl(i_{m+1}) - \text{weight}(< Ti_m, Ti_{m+1} >) = ve(i_m)$$





## 定理6.3（新）拓展

- 推论 1：假设边 $\langle T_i, T_j \rangle$ 属于AOE网，则有

$$ve(j) - ve(i) \geq \text{weight}(\langle T_i, T_j \rangle)$$

且如果 $\langle T_i, T_j \rangle$ 属于某一条关键路径，则

$$ve(j) - ve(i) = \text{weight}(\langle T_i, T_j \rangle)$$

- 由推论可知，活动 $\langle T_i, T_j \rangle$ 的最大可利用时间是 $vl(j) - ve(i)$ 。如果最大可利用时间等于边 $\langle T_i, T_j \rangle$ 的权值 $\text{Weight}(\langle T_i, T_j \rangle)$ ，则该活动是关键活动，该活动延期将导致整个工程延期。如果最大可利用时间大于 $\text{Weight}(\langle T_i, T_j \rangle)$ ，则该活动不是关键活动，其持续时间只要不超过最大可利用时间，就不会影响整个工程的进度。



## 定理6.3（新）的逆定理不成立

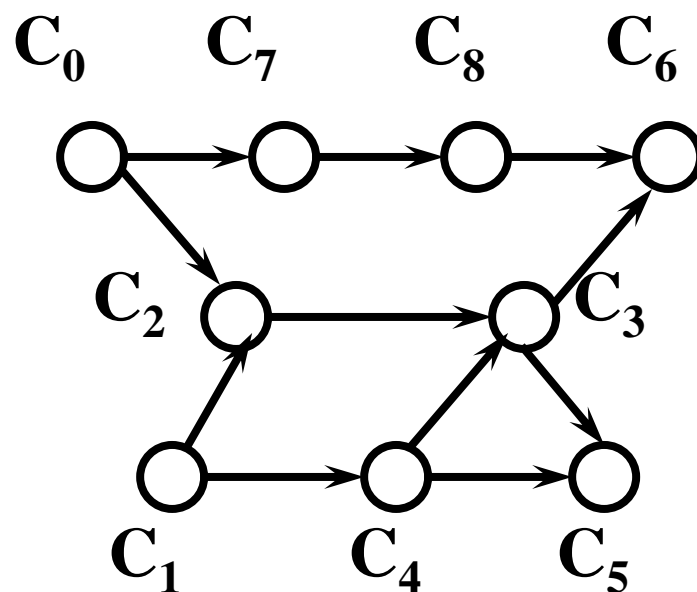
- 一条路径上的所有顶点都满足 $ve(i) = vl(i)$ ，则为关键路径；（错误）
- 因为，边 $\langle Ti, Tj \rangle$ 两端点都满足该等式，不能保证是边 $\langle Ti, Tj \rangle$ 关键活动。



# 点权图

## 计算机专业必修课程

课号	课名	先修	学期数
$C_0$	高等数学	无	3
$C_1$	程序设计基础	无	1
$C_2$	离散数学	$C_0, C_1$	2
$C_3$	数据结构	$C_2, C_4$	1
$C_4$	程序设计语言	$C_1$	1
$C_5$	编译技术	$C_3, C_4$	2
$C_6$	操作系统	$C_3, C_8$	2
$C_7$	普通物理	$C_0$	2
$C_8$	计算机原理	$C_7$	2





# 点权图关键路径处理方法

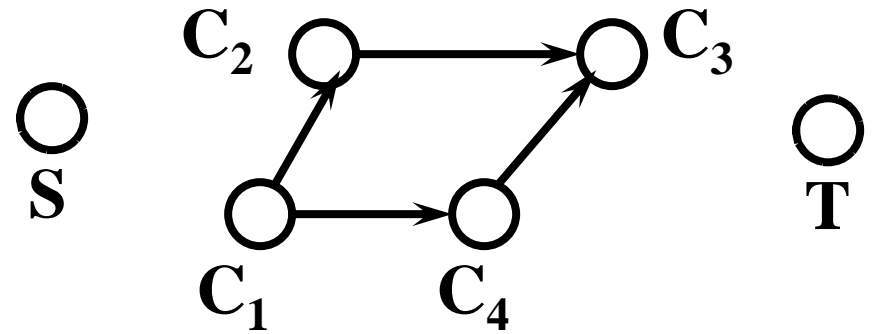
## □ 引入虚源虚汇（常用技巧）

## □ 关键路径处理方法

✓ 按点计算路径

✓ 点权转换为边权

- 每个点 $v$ 引入镜像点 $v'$ ，点权放到新增边 $\langle v, v' \rangle$ 上；原来的边 $\langle u, v \rangle$ 转换为 $\langle u', v \rangle$ 。
- 点权向后推到边上





# 关键路径小结

□ 关键路径的求解方式是**DAG**上的递推。

## □ 思考

- ✓ 如何求出一条关键路径（如序号排列最小）？
- ✓ 如何求出所有关键路径？
- ✓ 关键路径的条数？