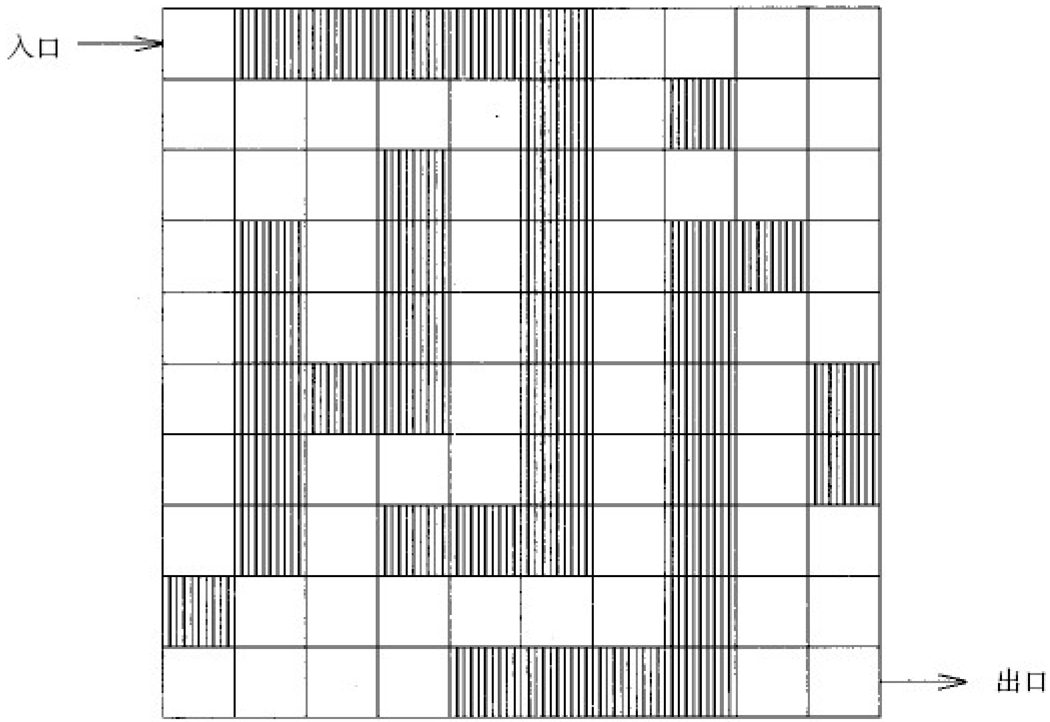
第三部分

图(网)结构的应用

犯罪团伙问题

- 警察抓到了n个罪犯，警察根据经验知道他们属于不同的犯罪团伙，却不能判断有多少个团伙，但通过警察的审讯，知道其中的一些罪犯之间相互认识，已知同一犯罪团伙的成员之间直接或间接认识。有可能一个犯罪团伙只有一个人。请你根据已知罪犯之间的关系，确定犯罪团伙的数量和每个犯罪团伙的罪犯。

迷宫问题



选址问题

- 选址问题，是指为一个和几个服务设施在一定区域内选定它的位置，使某一指标达到最优解。这类问题，在规划建设中经常可以碰到，这里所谓的服务设施，可以是某些公共服务设施，如医院，消防站，物流中心等。也可以是生产服务设施，如仓库，转运站等等。

选址问题示例

- 已知n个社区之间的交通图，现在要从这n个社区中选择一个社区建一所医院，问这所医院应建在哪个社区，能使所有的社区离医院都比较近（能使离医院最远的社区到医院最近）或 能使所有的社区到达医院的距离之和达到最小？

医院选址问题

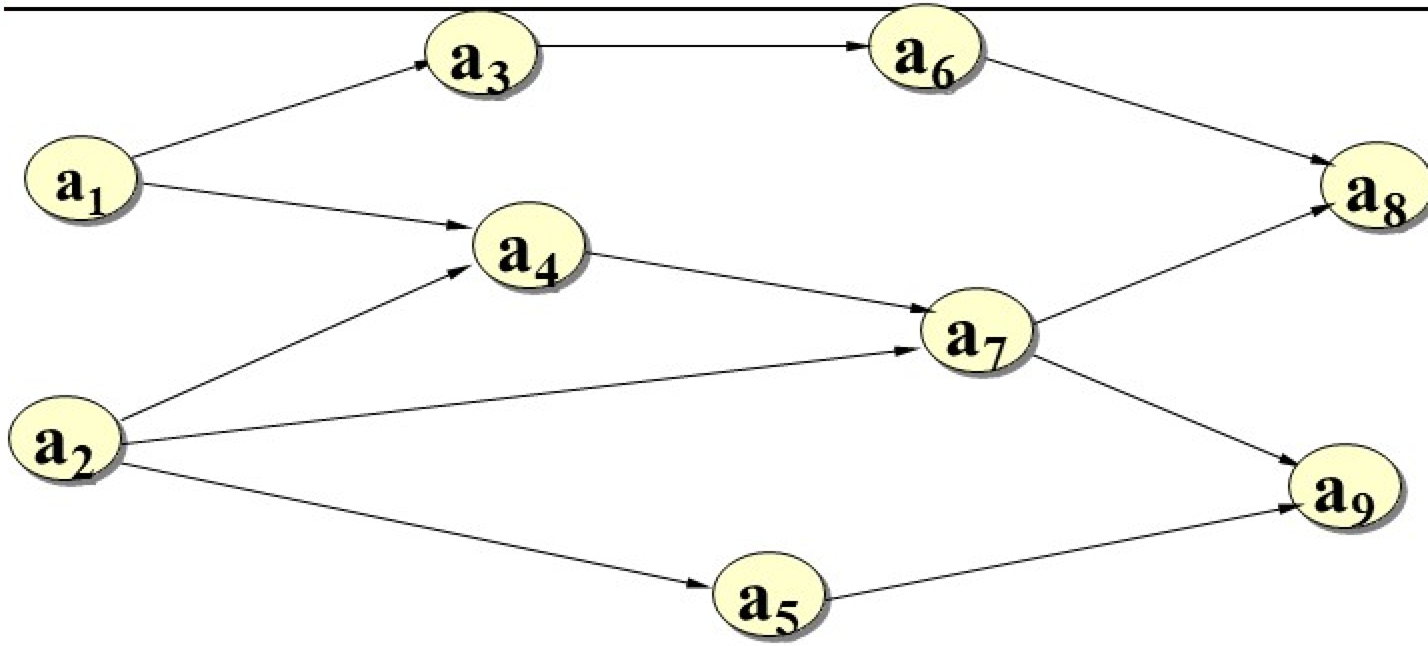
- n个社区之间的交通图用有向加权图表示，现在要从这n个社区中选择一个社区建一所医院，问这所医院应建在哪个社区，
- 能使所有的社区离医院都比较近（能使离医院最远的社区到医院最近）
- 或
- 能使所有的社区到达医院的距离之和达到最小

- 有向加权图:社区之间的交通,
  - 顶点：社区，边：社区之间的道路，权值：社区之间的距离
- Floyd算法

任务调度问题

- 一个工程项目由一组子任务构成，子任务之间有的可以并行执行，有的必须在完成了其它一组子任务后才能执行。
- 给出一个可行的任务调度方案
- 或 判定一个给定的任务调度是否可行。

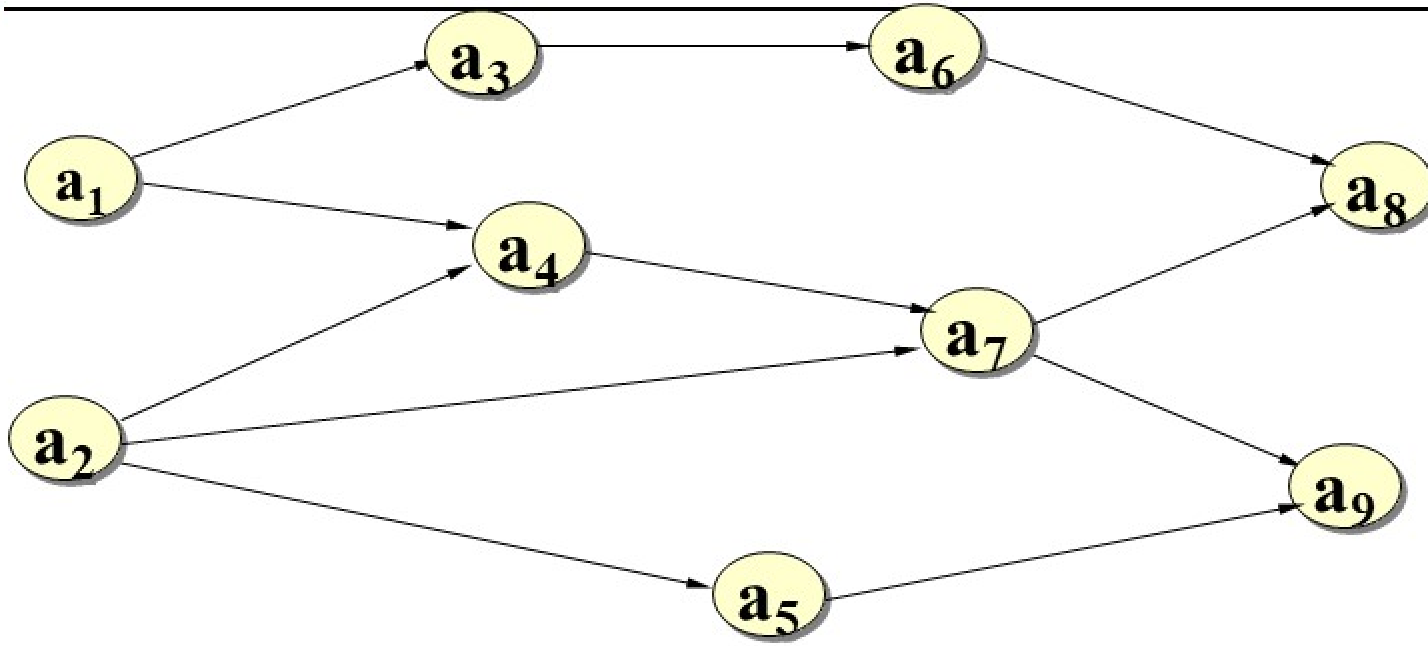
任务调度问题



- 使用**AOV**表示工程项目，
  - 顶点表示子任务，
  - 从$\mathbf{a_i}$到$\mathbf{a_j}$的有向边$(\mathbf{a_i}, \mathbf{a_j})$：

  子任务$\mathbf{a_i}$完成后，子任务$\mathbf{a_j}$才能开始执行；
- 拓扑排序

任务调度问题



- 思考：
  - 如果给出每个子任务的完成时间，如何计算该任务调度完成整个工程项目所需要的时间？

任务调度问题

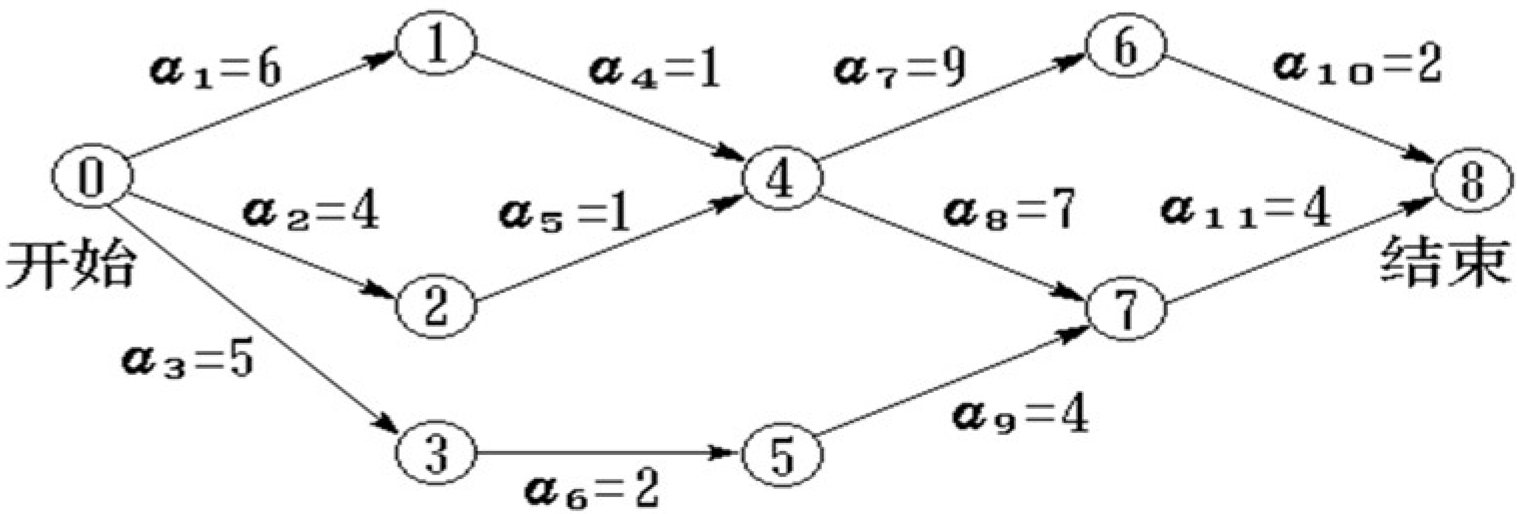
- 在任务调度问题中，如果给出每个子任务的完成时间，
  - 可以计算完成整个工程项目所需要的最短时间？
- 在所有的子任务中，有些任务即使推迟几天完成，也不会影响全局的工期；但是有些任务必须按时完成，否则真个项目的工期就要因此延误，这种任务就叫“关键活动”。
  - 如何确定工程中的“关键活动”？
  - 为缩短完成工程所需的时间，应当加快哪些活动?

用边表示活动的网络(AOE网络)

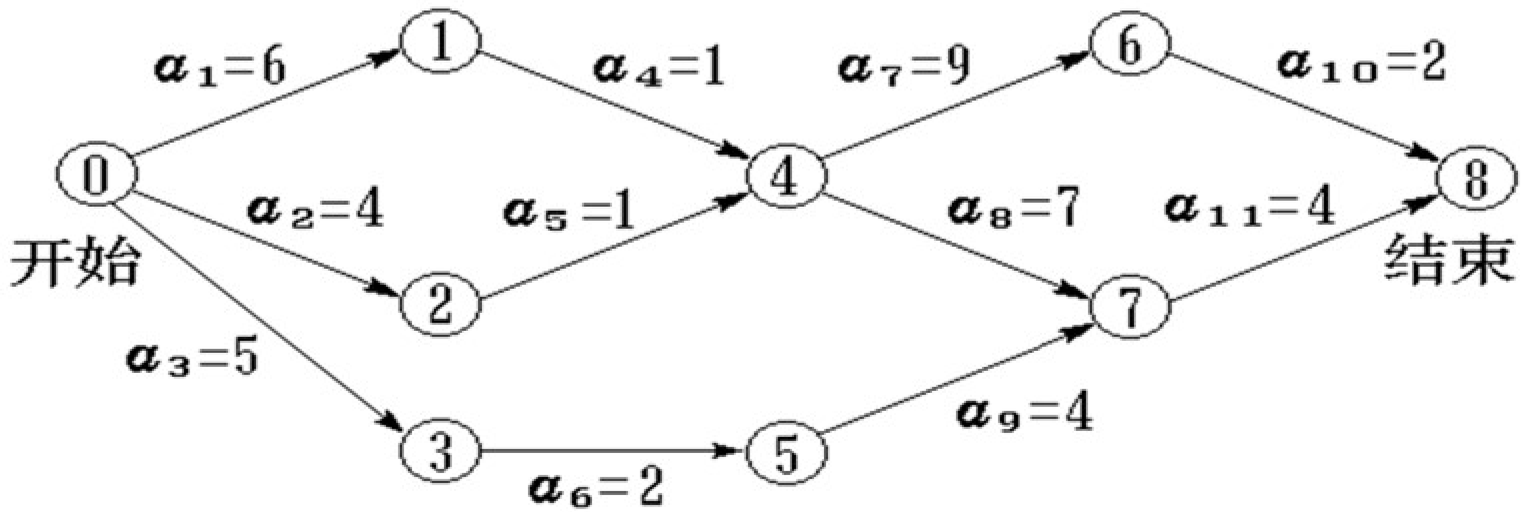
- 如果在无有向环的带权有向图中
  - 用有向边表示一个工程中的各项活动(Activity)
  - 用边上的权值表示活动的持续时间(Duration)
  - 用顶点表示事件(Event)
- 则这样的有向图叫做用边表示活动的网络，简称AOE (Activity On Edges)网络。
- AOE网络在某些工程估算方面非常有用。例如，计算：
  - (1) 完成整个工程至少需要多少时间(假设网络中没有环)?
  - (2) 为缩短完成工程所需的时间，应当加快哪些活动?

AOE网络

- 在AOE网络中，有些活动顺序进行，有些活动并行进行。
- 从源点到各个顶点，以至从源点到汇点的有向路径可能不止一条。这些路径的长度也可能不同。完成不同路径的活动所需的时间虽然不同，但只有各条路径上所有活动都完成了，整个工程才算完成。



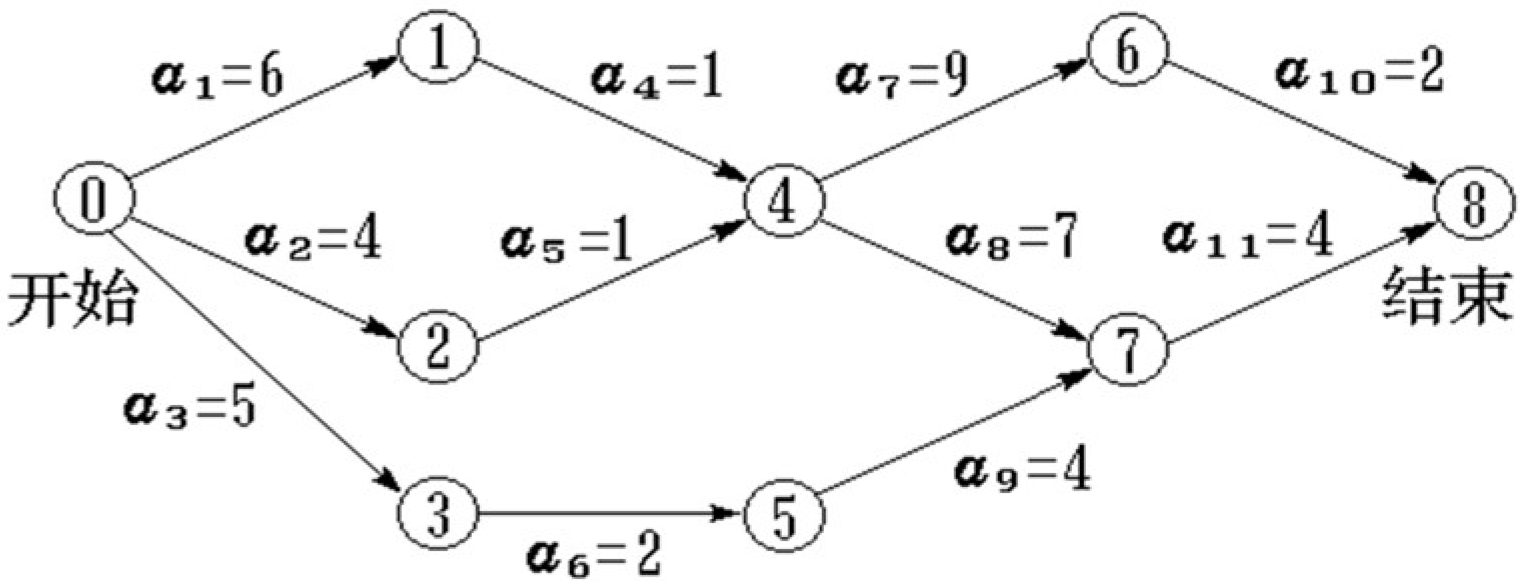
关键路径



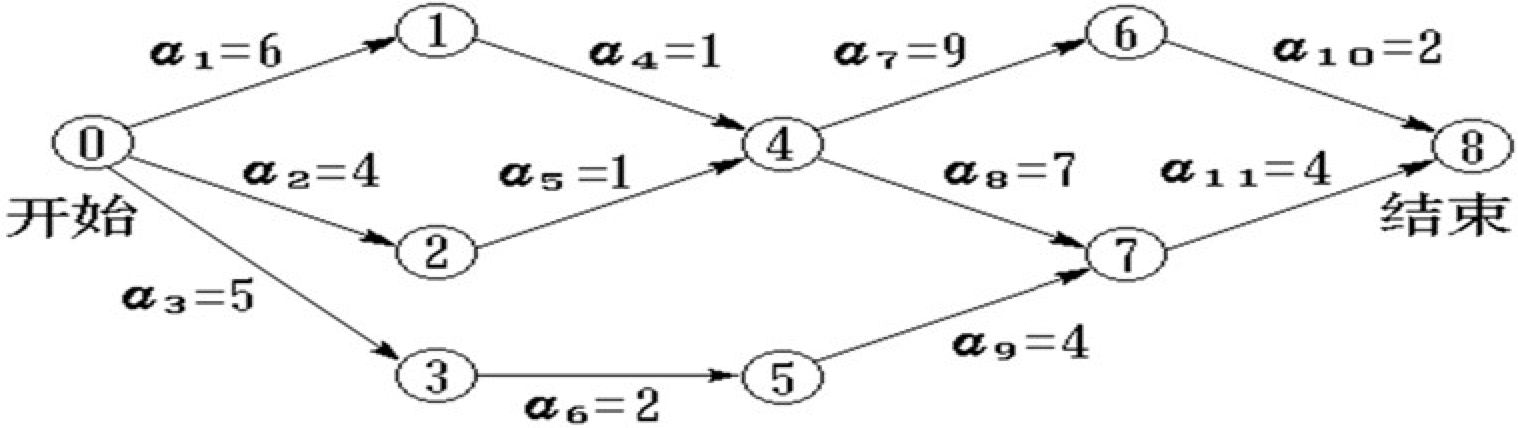
- 完成整个工程所需的时间取决于从源点到汇点的**最长路径长度**，即在这条路径上所有活动的持续时间之和。这条路径长度最长的路径就叫做**关键路径**(Critical Path)。

关键活动

- 要找出关键路径，必须找出关键活动，即**不按期完成就会影响整个工程完成的活动**。
- 关键路径上的所有活动都是关键活动。因此，只要找到了关键活动，就可以找到 **关键路径**

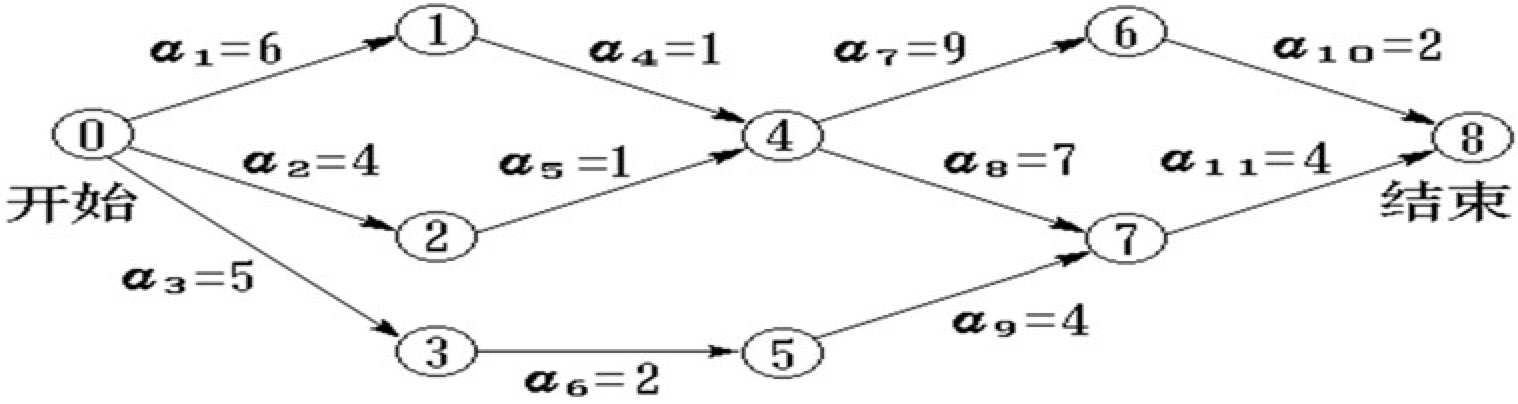


几个与计算关键活动有关的量



- 事件V_i的最早可能开始时间$Ve(i)$
  是从源点V_0到顶点V_i的最长路径长度。
  $Ve(1)=6$; $Ve(4)=7$; ……, $Ve(8)=18$
- 事件V_i的最迟允许开始时间$Vl[i]$
  是在保证汇点V_{n-1}在$Ve[n-1]$时刻完成的前提下，事件V_i的允许的最迟开始时间。
  $Vl(8)=18$; $Vl(6)=16$;

几个与计算关键活动有关的量



- 活动a_k的最早可能开始时间$e[k]$

  设活动a_k 在边$<V_i ,V_j>$上,则$e[k]$是从源点V_0到顶点V_i 的最长路径长度。因此,$e[k]=Ve[i]$。

- 活动a_k的最迟允许开始时间$l[k]$

  $l[k]$是在不会引起时间延误的前提下,该活动允许的最迟开始时间。

  $$l[k] = Vl[j] - dur(<i, j>)。$$

  其中，$dur(<i, j>)$是完成a_k所需的时间。

几个与计算关键活动有关的量

- 时间余量 $l[k]$ - $e[k]$

  表示活动a_k的最早可能开始时间和最迟允许开始时间的时间余量。

- $l[k] == e[k]$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 $e[k]$ 与 $l[k]$, 以判别是否 $l[k] == e[k]$.
- 为求得$e[k]$与 $l[k]$，需要先求得从源点V_0到各个顶点V_i的 $Ve[i]$ 和 $Vl[i]$。

计算$Ve[i]$, $Vl[i]$

- 求$Ve[i]$的递推公式
  - 从$Ve[0] = 0$开始，向前递推

$$Ve[i] = \max_{j}\{ Ve[j] + dur(< V_j, V_i >) \},$$

$$< V_j, V_i > \in S2, \quad i = 1, 2, \ldots, n\text{-}1$$

其中, $S2$是所有指向顶点V_i的有向边$< V_j , V_i >$的集合。

- 求$Vl[i]$的递推公式
  - 从$Vl[n\text{-}1] = Ve[n\text{-}1]$开始，反向递推

$$Vl[i] = \min_{j}\{ Vl[j] - dur(< V_i, V_j >) \},$$

$$< V_i, V_j > \in S1, i = n\text{-}2, n\text{-}3, \ldots, 0$$

其中, $S1$是所有从顶点V_i发出的有向边$< V_i , V_j >$的集合。

- 这两个递推公式的计算必须分别在**拓扑有序**及**逆拓扑有序**的前提下进行。

计算$e[k]$, $l[k]$

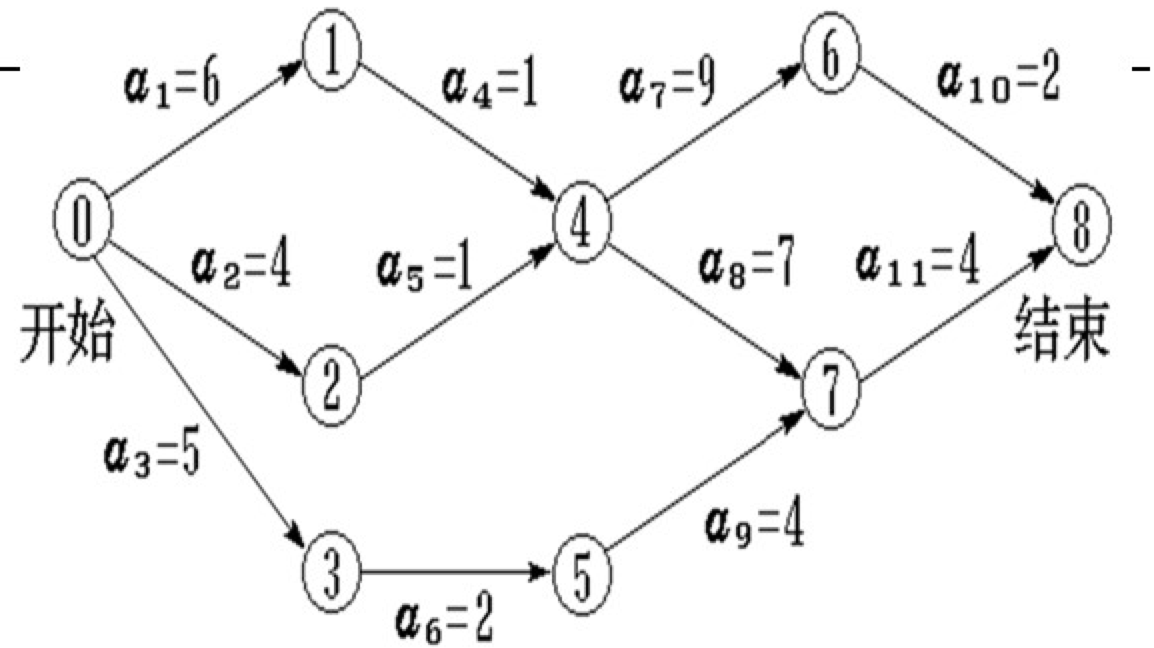
- 设活动$a_k(k=1,2,\ldots,e)$在带权有向边$<V_i,V_j>$上, 它的持续时间用$dur(<V_i,V_j>)$表示，则有

  $e[k]=Ve[i]$;

  $l[k]=Vl[j]-dur(<V_i,V_j>);k=1,2,\ldots,e$。

- 计算关键路径时,可以一边进行拓扑排序一边计算各顶点的$Ve[i]$。
- 为了简化算法，假定在求关键路径之前已经对各顶点实现了拓扑排序，并按拓扑有序的顺序对各顶点重新进行了编号。算法在求$Ve[i], i=0,1,\ldots,n-1$时按拓扑有序的顺序计算，在求$Vl[i],i=n-1,n-2,\ldots,0$时按逆拓扑有序的顺序计算。

事件	*Ve*[*i*]	*Vl*[*i*]
V_0	0	0
V_1	6	6
V_2	4	6
V_3	5	8
V_4	7	7
V_5	7	10
V_6	16	16
V_7	14	14
V_8	18	18



边	<0,1>	<0,2>	<0,3>	<1,4>	<2,4>	<3,5>	<4,6>	<4,7>	<5,7>	<6,8>	<7,8>
活动	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}
e	0	0	0	6	4	5	7	7	7	16	14
l	0	2	3	6	6	8	7	7	10	16	14
l - e	0	2	3	0	2	3	0	0	3	0	0
关键	是			是			是	是		是	是