DAG and Critical Path Method

March 27th 2019

Methods Of Mathematical

Abstract

- Some Practical Problems in the area of process planning.
- DAG(definition,some common results on DAG) (eg1)
- Topological Sort for DAG (eg1)
- AOV model and its algorithm (eg1)
- AOE model and its algorithm (eg2)
- DAG and Frankl’s conjecture

举例1

- Eg1
- 问题：
- 给出一个信科本科生的课程先修要求提示清单，试据此排出一个合理的修课顺序，使得先修的课程总在旧修的课程之前。

课程编号	课程名称	先修课程
C1	程序设计基础	无
C2	离散数学	C1
C3	数据结构	C1,C2
C4	汇编语言	C1
C5	语言的设计和分析	C3,C4
C6	计算机组成原理	C11
C7	编译原理	C5,C3
C8	操作系统原理	C3,C6
C9	高等数学	无
C10	线性代数	C9
C11	普通物理	C9
C12	数值分析	C9,C10,C1

举例2

- Eg2 通常在一个工程中一些活动可以并行地进行，另一些活动则需要等待前面所有的活动完成后才能开始，那么这些活动以及活动完成时立即发生的事件可以画出一个有工序先后顺序的流程图来。
- 问：

显然完成整个工程的最短时间肯定是最长的那条流程图（就像木桶的盛水量取决于最短的那个木板）。给定一个流程。那么我们如何求出工程的最长时间，又如何给出一个合理地优化的方案呢？

理论分析（eg1）

- 一、拓扑排序（Topological Sort）
  - 偏序：若集合 X 上的关系 R 是传递的、自反的、反对称的，
  - 则称 R 是集合 X 上的偏序 关系。（仅有部分元素可比较大小(或先后)）
  - 全序：若关系 R 是集合 X 上的偏序关系，如果对于每个 x,
  - y 属于 X，必有 x R y 或 y R x，则称 R 是集合 X
  - 上的全序关系。（所有元素可比较大小(或先后)）

  - 拓扑排序：如果有一个序列 $A = a_1,a_2,a_3............a_n$ 是集合
  - X 上的元素的一个序列，且当 $i < j$ 时，
  - （a_i , a_j）属于 R，
  - 则称 A 是相对于 R 拓扑排序的。（即把偏序关系转化为全序关系的过程）

理论分析（eg1）

- 二、无圈有向图的基本事实：
- 1 任何一个无圈有向图具有一个零入度的顶点和一个零出度的顶点

2 任何一个无圈有向图含有顶点的一个无圈序（拓扑排序）。

3 设D是一个无圈有向图有且仅有一个入度（或出度）为零的顶点x或（y）。则对每一个顶点v属于V（D），在D中存在一条（x，v）路和一条（v，y）路。

4 命题2中的存在一个算法，其复杂度能够控制在时间O(|v(D)|+A(D)|)

模型构建（eg1）

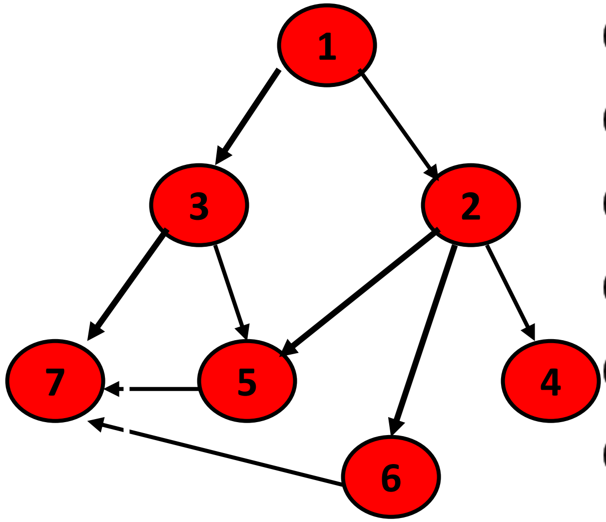
- 要解决eg1中的问题实际上可以令每个课程对应一个顶点，Ai须先于Aj修，可以用一条Ai指向Aj的弧表示，这样构成了一个DAG，要列出合理的修课顺序即使要给出一个顶点的拓扑排序。

图的邻接矩阵：

- 定义：有向图邻接矩阵中第i行非零元素的个数为第i个顶点的出度，第i列非零元素的个数为第i个顶点的入度，第i个顶点的度为第i行与第i列非零元素个数之和。
- 作用之一:用邻接矩阵表示图，很容易确定图中任意两个顶点是否有边相连。
- 性质：1.一个有向无环图的邻接矩阵所有特征值为0
- 2. Eric W. Weisstein 发现： n*n 阶0-1矩阵的特征值是正实数时，矩阵总数的序列和包含n个标记顶点的有向无环图的个数序列相等：

解决eg1

- 利用邻接矩阵写一个无圈有向图拓扑排序的算法：



$$\begin{matrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} \qquad \begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

算法的执行步骤：

1、找到全为零的第j 列，输出 j

2、将第j 行的全部元素置为零

3、找到全为零的第k 列，输出 k

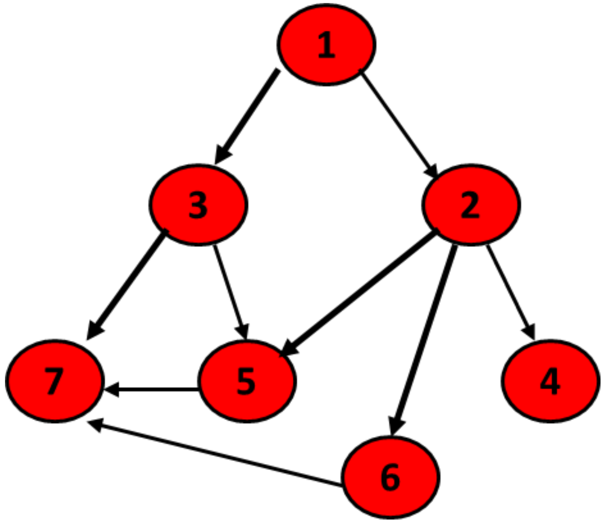
4、将第 k 行的全部元素置为零

……………………

反复执行3、4；直至所有元素输出完毕。

解决eg1

```
Status  Topologicalsort（ ALGraph G）
{   findinDegree（G， indegree）；
     Initstack（S）；
     for （i = 0; i < G.vexnum; ++i）
          if （! Indegree [ i ]）Push（S， i）；
    count = 0;
    while （！ StackEmpty（S））
    {    Pop（S， i）；Printf（i， vertices[ i ].data）；++count;
         for （p=G.vertices[i]. firstarc; p; p=p->nextarc）；
          { k = p->adjnexr;
             if （!（- - indegree [ k ]））  Push（S, k）；}
     }
     if （count < G.vexnum）return ERROR;
     else return OK;
} //  end
```



AOV与AOE模型

- 上述由实际问题转换而来的无圈未赋权的DAG就叫AOV模型。但实际生产生活中我们所遇到的往往不止仅有逻辑关系，两个事务间的逻辑关系可能还存在一定的加权，比如往返两地的时间和工程耗时。

- 所以我们还要引入给有向图的边加权赋值的AOV模型也就是AOE模型，特别的，为了能方便分析我们假设它只有**唯一**一个入度为0的点，**唯一**一个出度为0的点，没有特别说明我们假设它已按拓扑排列排好为A0，A1，A2…An-1，且可以用角标j表示相应的点。

Eg2

- Eg2 通常在一个工程中一些活动可以并行地进行，另一些活动则需要等待前面所有的活动完成后才能开始，那么这些活动以及活动完成时立即发生的事件可以画出一个有工序先后顺序的流程图来。
- 问：　　　显然完成整个工程的最短时间肯定是最长的那条流程图（就像木桶的盛水量取决于最短的那个木板）。给定一个流程。那么我们如何求出工程的最长时间，又如何给出一个合理地优化的方案呢？

模型构建（eg2）

- 构建AOE网络：其中定义 结点为事件，有向边的指向表示事件的执行次序。单位是时间（时刻）。有向边定义为活动，它的权值定义为活动进行所需要的时间。
  - 源点：表示整个工程的开始点，也称起点。
  - 汇点：表示整个工程的结束点。
  - 事件结点：单位时间，表示的是时刻。
  - 活动（有向边）：它的权值定义为活动进行所需要的时间。方向表示起始结点事件先发生，而终止结点事件才能发生。

理论分析（eg2）

- 为了解决问题3，我们给出如下定义：
- 定义Ai到Aj所有可能赋权有向路径长度**数值**的集合记为dut（Ai，Aj），若i，j属于有向图的边集合，其赋权值记为d（Ai，Aj）。
- 1 在赋权无圈有向图中，设A0为**唯一**一个入度为0的点，An-1为**唯一**一个出度为零的节点。则任意节点i（对应一个事件发生）相对于起始点A0的最早发生时间Ve（i）=Max｛dut（A0，Ai）｝
- 2 在赋权无圈有向图中，设A0为唯一一个入度为0的点，An-1为唯一一个出度为零的节点。则任意节点i （对应一个事件发生）相对于结束点An-1的最迟发生时间Vl（i）=Ve(n-1)-Max{dut(Ai,An-1)}
- 3 在赋权无圈有向图中，若对一个节点i，1、2中定义的Ve（i）=Vl（i)
- 相等，则称之为关键节点。

理论分析（eg2）

- 事实一：
- A（n-1）点的最早发生时间Ve与最迟发生时间Vl相等，因此它是一个关键点。
- 事实二：
- A0点最早发生时间为0，最迟发生时间也为0，因此它也是一个关键点。
- 性质一：
- Aj的最早发生时间Ve（j）=Max{Ve（i）+max{dut（i，j）}}其中（i，j）属于该有向图的边集。
- 推论：Ve（j）=Max{Ve（i）+d（i，j）}

理论分析（eg2)

- 性质二：
- Aj的最迟发生时间Vl（j）={Vl(k)-max{dut(Aj,Ak)}}min其中(j,k)属于这个有向图的边集。
- 推论：Vl（j）= {Vl（k）-d（Aj，Ak）} min其中（j，k）属于这个有向图的边集。
- 性质三：
- 这个图中所有关键点以及关键点间的那些边构成的赋权有向路（不唯一）正是我们要找的最长路。（若含“不关键的点”那么Max {dut（A0，Ai）+Max{dut(Ai,An-1)}<Ve(n-1)=Vl(n-1),与路的最长性矛盾；故这条路中含且只含关键点；反之，若都含的是关键点也有d（A0-A（i1）…—An-1）=Ve（n-1））

解决eg2

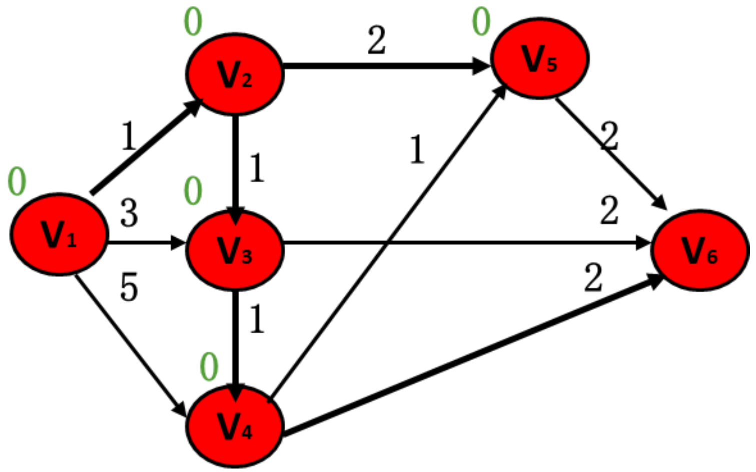
- 算法处理：根据以上的递推公式我们不难写出要求一个流程图关键路径及最长路径的算法。
- 第一大步：求出每个事件的最早发生时间（包括了Ve（n-1））
- 第二大步：求出每个事件的最迟发生时间
- 第三大步：写出关键点
- 第四大步：找出可能的关键路径，计算工作时间（其实在第一大步已经完成，重点是要找出关键路径，即找出“木桶的短板”，好给出优化方案。）

解决eg2

- 第一大步：求出每个事件的最早发生时间（包括了Ve（n-1））
- 根据递推关系和拓扑排序算法给出求事件结点的最早发生时间的计算机执行步骤：
- 1、设每个结点的最早发生时间为0，将入度为零的结点进栈。
- 2、将栈中入度为零的结点V取出，并压入另一栈，用于形成逆向拓扑排序的序列。
- 3、根据邻接表（邻接矩阵也可）找到结点V的所有的邻接结点，将（结点V的最早发生时间 + 活动的权值） 得到的和同邻接结点的原最早发生时间进行比较；如果该值大，则用该值取代原最早发生时间。另外，将这些邻接结点的入度减一。如果某一结点的入度变为零，则进栈。
- 4、反复执行 2、3；直至栈空为止。

解决eg2

- Status Topologicalsort（ ALGraph G，Stack &T）
- { FindinDegree（G，indegree）；
- // 对各顶点求入度，建立入度为零的栈 S，
- Initstack（T）； count = 0;
-    ve [0 .. G.vexnum - 1] = 0;
- while （！ StackEmpty（S））
- { Pop（S，j）;Push（T，j）; ++count;
-     for （p=G.vertices[i]. firstarc; p; p=p->nextarc）；
-     {   k = p->adjnexr;
-         if （!（- - indegree [k]））Push（S, k）；
-         if （ve[j]+ *（ p->info）> ve[k]）
-                 ve[k] = ve[j] + *（ p->info）; }
- }
-      }
-     if （count < G.vexnum）return ERROR;
-     else return OK;
- } // 栈 T 为求事件的最迟发生时间的时候用。



解决eg2

- 第二大步：求出每个事件的最迟发生时间：
- 根据递推关系和逆向拓扑排序算法可以给出求事件结点的最迟发生时间的执行步骤：
- 1、设每个结点的最迟发生时间为收点的最早发生时间。
- 2、将栈中的结点V取出。
- 3、根据逆邻接表找到结点V的所有的起始结点，将（结点V的最迟发生时间 - 活动的权值得到的差同起始结点）的原最迟发生时间进行比较；如果该值小，则用该值取代原最迟发生时间。
- 4、反复执行 2、3；直至栈空为止。

回顾Eg2

- 以上建模和分析过程的重点在于：

（1） 完成整个工程至少需要多少时间；

- （2） 哪些活动是影响工程的关键。

以上分析后采用的算法又叫关键路径法：

- 1956年，美国杜邦公司提出关键路径法，并于1957年首先用于1000万美元化工厂建设，工期比原计划缩短了4个月。杜邦公司在采用关键路径法的一年中，节省了100万美元。
-

DAG and Frankl’s conjecture

- DAG 不仅可以用于刻画一些实际生活问题，事实上很多和顺序有关的数学问题都可能与DAG发生联系，也可以用DAG这种模型来刻画，从某种意义上DAG可以说是一种“图形化”的格系统。

1.A brief introduction to union-closed family conjecture .

● In combinatorial mathematics, the **union-closed sets conjecture** is an elementary problem,
posed by Péter Frankl in 1979 and still open.

● A family of sets is said to be *union-closed* if the union of any two sets from the family remains in the family.

● The conjecture states:
For every finite union-closed family of finite sets, other than the family containing only the empty set,
there exists an element that belongs to at least half of the sets in the family.

1、A brief introduction to union-closed family conjecture .

●Let U be a finite set and F a family of nonempty subsets of U which is closed under unions.

Question: Can you prove there is an element of U which is in at least half the sets of F?

●An equivalent form of the conjecture is that for any family of proper subsets of U closed under intersection, there must exist some element of U that belongs to at most half of the sets of F.

● The lattice formulation 1: Conjecture 1. Let L be a finite lattice with at least two elements. Then there is a join-irreducible element a with $|[a)| \leq 1/2\ |L|$.

● The Graph formulation 1: Conjecture 2. Let G be a finite graph with at least one edge. Then there will be two adjacent vertices each belonging to at most half of the maximal stable sets.

● The Graph formulation 2: Conjecture 3. Let G be a finite bipartite graph with at least one edge. Then each of the two bipartition classes contains a vertex belonging to at most half of the maximal stable sets.

1、A brief introduction to union-closed family conjecture .

● Some results

1989Sarvate, D. G.; Renaud, J.-C. ,the case there exists an element of F of size at most 2

1994Lo Faro, Giovanni, the case $|F| \le 36$

1995Morris, Robert, the case $|U| \le 9$

1999Wojcik, Piotr, at least N/log2N

2010 Ivica Bošnjak, Petar Marković |U|<=11

● 2000 Abe, Tetsuya each lattice can be translated into a union-closed set family, and each union-closed set family can be translated into a lattice

the conjecture is that in any finite lattice there exists an element x that is not the join of any two smaller elements, and such that the number of elements greater than or equal to x totals at most half the lattice

● 2017 Henning Bruhn,Oliver Schaudt almost all random bipartite graphs

Study main ideas、study methods and plan.

- Main ideas and methods:
- 1Use Directed Acyclic Graph(DAG) to study The Union-closed family conjecture.

- Plan:
- Some results I find out:
- 1 each Union-closed family can be translated into a DAG.
- 2 We proof that for each union-closed family,(|F|=n)there exists a subfamily which |F`|=k(1<=k<=n)
- 3We proof some special cases which this conjecture is true.
- 4 We proof this conjecture is equal to a combination identical equation about DAG

- References

- [1] Abe, Tetsuya, Strong semimodular lattices and Frankl's conjecture, Algebra Universalis
- 44 (2000), no. 3-4, 379-382.
- [2] Lo Faro, Giovanni, Union-closed sets conjecture: improved bounds, J. Combin. Math.
- Combin. Comput. 16 (1994), 97-102.
- [3] Morris, Robert, FC-families and improved bounds for Frankl's conjecture, European
- Journal of Combinatorics 27 (2006), no. 2, 269-282.
- [4] Poonen, Bjorn, Union-closed families, Journal of Combinatorial Theory, Series A 59
- (1992), no. 2, 253-268.
- [5] Reinhold, Jrgen, Frankl's conjecture is true for lower semimodular lattices, Graphs
- Combin. 16 (2000), no. 1, 115-116.
- [6] Sarvate, D. G.; Renaud, J.-C. On the union-closed sets conjecture, Ars Combin. 27
- (1989), 149-153.
- [7] Wojcik, Piotr, Union-closed families of sets, Discrete Math 199 (1999), 173—182.

- Thank You!