第 34 卷 第 9 期 计 算 机 工 程 2008 年 5 月
Vol.34 *No.9* **Computer Engineering** **May 2008**

• 软件技术与数据库 • 文章编号：1000—3428(2008)09—0106—03 文献标识码：A 中图分类号：TP311.4

一种求解关键路径的新算法

王明福

(深圳职业技术学院软件工程系，深圳 518055)

摘 要：通过定义节点编码图概念，提出一种不需要拓扑排序的求解关键路径的新算法。该算法扩充图的邻接表的存储结构，使图的存储与算法求解过程共享同一存储空间。从图的源节点开始，用加权取极大运算规则，广度优先递归对图中所有节点进行编码。编码图生成后，利用反向搜索求出从源点到汇点的所有关键路径及长度。该算法比现有算法更简单直观，所需的存储空间更小，算法时间复杂度降低到 $O(n+e)$，优于现有算法的 $O(n^2)$。

关键词：编码图；关键路径；AOE 网；广度优先搜索；时间复杂度

New Algorithm for Finding Critical Paths

WANG Ming-fu

(Dept. of Software Engineering, Shenzhen Polytechnic, Shenzhen 518055)

【Abstract】 A new algorithm for finding the critical paths is proposed by using coding graph and without topological sort. By extending the data structure of the adjacency lists in the graph, the graph is stored in the same storage space with the algorithm. Beginning from the source node of the graph, by using the rule of getting the maximum number in the weighing calculation and breadth-first search, it encodes all nodes in the graph recursively. That recursively accessing the adjacent node and starting from the current node is the process of re-estimation about preceding node set and the length value from the adjacent node to the source node. After creating the coding graph, by inversing search, it can find all critical paths and the length from destination node to source node. Compared with the traditional algorithms, the algorithm proposed is simpler, more understandable and needs less storage space. The time complexity is $O(n+e)$, which is lower than $O(n^2)$ of the traditional algorithm.

【Key words】 coding graph; critical paths; Activity on Edge(AOE) network; breadth-first search; time complexity

1 概述

网络计划技术是一种组织生产和进行计划管理的科学方法，通常把工程计划表述为带权的有向无环图，以节点表示事件，以有向边表示活动，边上的权值表示该活动的持续时间，则此带权的有向图称为用边表示“活动”的网，简称 AOE 网(Activity on Edge network)。为了进行人力、物力的调度和分配、缩短工期，必须找出影响工程进度的关键活动，这就是关键路径的求解问题。

传统求解关键路径的算法[1]存在以下不足：(1)算法需要分别求出所有事件的最早发生时间和最迟发生时间以及每项活动的最早开始时间和最迟开始时间，然后判断哪些活动是关键活动，算法过程复杂。(2)算法执行完毕，只能知道哪些是关键活动，却不能统计从源点到汇点的关键路径数，也不能将每条关键路径输出。文献[2]对传统算法进行了一些改进，在广度优先搜索的基础上，采用图的十字链表结构形式，不需要拓扑排序即可求出所有关键活动。虽然无须进行拓扑排序，但采用十字链表作为存储结构会比较复杂，而且需要对图进行 3 次广度优先搜索才可以输出所有关键活动，且没能把所有的关键路径输出。文献[3]在深度优先搜索的基础上，求出从源点到汇点的所有路径，经过分析比较后找出最长的路径，从而求取关键路径。在求解过程中需要进行多次递归回溯，算法的执行效率较低。文献[4]针对上述不足提出了一种在广度优先搜索基础上，利用优先队列，结合动态规划思想实现的求解关键路径的算法。使算法的时间复杂度降低为 $O(n+e)$，然而，由于算法采用图的邻接表形式，以自底向上的方式计算最优值，并在此过程中记录一些有用信息，因此需要额外的存储空间，同时，采用一个按照入度值为优先级的优先队列需要排序。在输出所有关键路径时，需要将每个节点信息转换为一个二维数组，因此，算法是以空间为代价换取时间。

本文突破传统的求解关键路径算法思想，通过引入节点编码图概念，提出了一种新算法。算法具有如下特点：(1)能高效求出图中从源点到汇点的所有关键路径、关键活动和长度。(2)简单、直观，与现有的算法相比，其时间复杂度降低了一个数量级。(3)省去了求解过程所需要的临时存储空间，使图的存储、求解过程和结果数据存储共享同一空间，存储空间消耗小。

2 基本概念与理论基础

2.1 基本概念

定义 1 给定有向加权图 $G=<V, E, W>$，设 $v_0,v_1,\cdots,v_n\in V$，对应边(或弧) $e_1, e_2,\cdots, e_m\in E$，其中，$e_i$ 是连接节点 v_{i-1}, v_i 的有向边，对应边的权 $w_{01},w_{12},\cdots,w_{(m-1)\,m}\in W$；序列 $v_0,\ v_1,\cdots,\ v_m$ 称为 v_0 到 v_m 的路径；$w_{01}+w_{12}+\cdots+w_{(m-1)\,m}$ 称为该路径的长度。

定义 2 设 $G = <V, E, W>$ 为 n 阶有向带权图，满足：(1)G

基金项目：粤港关键领域重点突破项目(06KJcd001)；深圳职业技术学院科技基金资助项目(03KJc054)

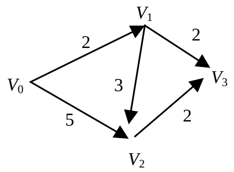
作者简介：王明福(1956－)，男，副教授、硕士，主研方向：计算机图形学，算法设计与分析，网络优化算法

收稿日期：2007-06-20 **E-mail：**wmf@oa.szpt.net

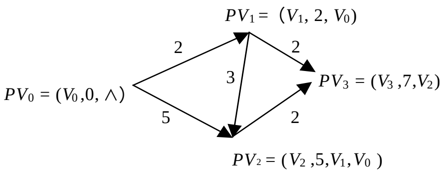
是一个简单无环图；(2)有一个节点入度为 0，称为源点，有一个出度为 0 的节点，称为汇点；(3)边$<v_i, v_j>$带的权记为w_{ij}，G称为AOE网。在AOE网中求关键路径就是求从源点到汇点的一条最长路径。

定义 3 设G=$<V,E,W>$是一个AOE网，记节点$V=\{V_i \mid i=0,1,2,\cdots,n\text{-}1\}$，则图$G$对应的源点$V_0$的编码图(简称$CG(V_0)$)定义为$CG(V_0)=<PV, E, W>$，其中，$PV=\{PV_i=(V_i, L_i, V_{ip}) \mid i=0,1,2,\cdots,n\text{-}1\}$；$L_i$称为$CG(V_0)$的节点$PV_i$标量，取距源点$V_0$最长路径的长度值(如无路径取 0 值)；$V_{ip}$是最长路径中当前节点$V_i$的前序节点集合。显然，AOE网与$CG(V_0)$的真子集同构。

例如，图 1(b)是图 1(a)中源点V_0的编码图。



(a)带权有向图G_1



(b)编码图$CG_1(V_0)$

图 1　图G_1及源点V_0的编码图$CG_1(V_0)$

根据编码图$CG_1(V_0)$的构造性定义，从编码图$CG_1(V_0)$的汇点PV_3出发，利用反向搜索，直到$CG_1(V_0)$源节点PV_0，即可求出所有关键路径。并且在汇点$PV_3 = (V_3, L_3, V_{3p})$中，$L_3$即为从$V_0$到$V_3$的关键路径长度值。

2.2　性质定理

定理 1 设$G=<V,E,W>$是一个AOE网，对源点V_0，存在唯一编码图$CG(V_0)=<PV,E,W>$。

证明：存在性显然成立。因为对任一节点V_i，如果有一条到V_0的关键路径，那么根据传统关键路径算法[1]即可求出关键路径及其长度，设某一条关键路径为$(V_0,\cdots, V_f, V_i)$，长度为L，取$PV_i=(V_i, L_i, V_{ip})=(V_i,L,\{V_f\})$即可；如果无路径，取$(V_i,L_i, V_{ip})=(V_i, 0,\wedge)$即可。

再证唯一性。首先，L_i唯一是因为关键路径长度是唯一的。另外，由于V_{ip}是所有的前序节点，如果有 2 组前序节点集$\{Vp1\}$和$\{Vp2\}$，取并集$V_{ip}=\{Vp1\}\cup\{Vp2\}$，又因为$V$是有限集，所以其子集$V_{ip}$也是有限集，唯一性得证。

定理 2 设$G=<V, E, W>$是一个AOE网，在源点V_0的编码图$CG(V_0)$中，设L是源点V_0到V_i的一条关键路径，若L经过中间节点V_k，且$L= L_1\cup L_2$，则L_1是V_0到V_k的最长路径，L_2是V_k到V_i的最长路径，即问题的最优解包含着子问题的最优解。

证明：不失一般性，反设L_1不是V_0到V_k的关键路径，Q_1是V_0到V_k的一条关键路径，那么在路径Q_1和L_1中至少存在一个不同的节点V_t，且路径Q_1的长度大于路径L_1的长度，显然，$L'=Q_1\cup L_2$是从V_0到V_i经过V_k的一条路径，其长度值大于路径L的长度，这与L是关键路径前提相矛盾，因此，结论得证。

定理 3 在V_0的编码图$CG(V_0)$中，从节点PV_i出发，利用反向搜索，如果L_i=0，则无路径；否则，最多经n-1(n是图G的节点数)步一定能达到源点V_0，并且求得节点序列V_i, V_{i1}, $V_{i2},\cdots,V_{im}, V_0$，其为源点$V_0$到节点$V_i$的一条关键路径。

证明：若结论不成立，那么在路径序列中，至少存在 1 个重复出现的节点V_k，与G为简单无环有向加权图矛盾，因此，最多经n-1 步一定能达到源节点V_0。根据编码图$CG(V_0)$的构造和定理 2，求得节点序列$V_i,V_{i1}, V_{i2},\cdots,V_{im} ,V_0$，其为源节点$V_0$到节点$V_i$的一条关键路径。

3　求解关键路径算法

3.1　数据结构

为了使图的存储与算法求解过程共享同一存储空间，使编码图$CG(V_0)$的存储无需任何额外开销，对传统图的邻域表结构形式进行扩展，使得对图G的邻域表的某些域进行修改，便生成编码图，存储结构设计如下：

(1)表头节点表，由所有表头节点以顺序结构(向量)的形式存储，以便随机访问任一节点的边链表。

```
typedef struct CodeGraph{//定义编码图表头节点结构
struct fvex *fnext;       //指向关键路径中的前序节点的指针
   int    L;              //存储从源点到该节点的关键路径长度
   struct node *next;     //指向边表节点的指针
   }* CODEGRAPH;
```

(2)边表，由表示图中节点间邻接关系的边表组成。

```
typedef struct node {     //定义边表节点结构
   int    nodeno;         //边所指向的节点编号
   int    weight;         //边的权值
   struct node *next;     //指向下一条边的指针
}* NODE;
```

边链表中节点的结构如图 2 所示。

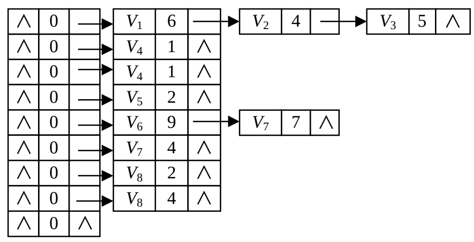


图 2　G_2的扩展邻接表表示法

(3)前序节点表，由指向下一前序节点的链域和前序节点序号组成。

```
typedef struct fvex {     //定义前序节点结构
      struct fvex *next;  //指向前序节点的指针
int    vexno;             //节点编号
}* FVEX;
```

编码图$CG_2(V_0)$如图 3 所示。

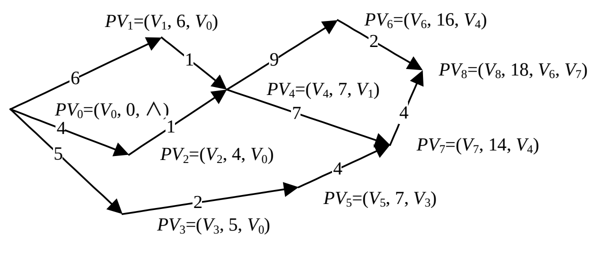


图 3　编码图$CG_2(V_0)$

3.2　算法描述

编码图的构造算法是一种递推算法，引入队列，对图G进行广度优先编码。从源点V_0开始，依次编码其邻接点，然后从这些邻接点出发依次编码它们的邻节点，并使“先被编码的节点的邻节点”先于“后被编码的节点的邻节点”被编

码，直到结束。算法 1 描述如下：

(1)初始化：队列初始化为空，AOE 网中所有节点到源点的关键路径长度 L 为 0，前序节点指针域为空。

(2)将源点V_0入队列。

(3)如果队列非空，则出队首节点V_i，并将节点V_i的L_i转移给所有邻节点，再将所有编码修改过的邻节点入队列。不失一般性，设当前出队节点是V_i，与V_i的K个单向邻节点是$(V_{j1}, L_{j1}, V_{j1p}), (V_{j2}, L_{j2}, V_{j2p}),\cdots, (V_{jk}, L_{jk}, V_{jkp})$，对应邻接边的权值分别为$w_{j1}, w_{j2}, w_{j3},\cdots, w_{jk}$，对邻节点$(V_{jm}, L_{jm}, V_{jmp})(m=1,2,3,\cdots,k)$做权值转移，取极大映射$\xi$：

$$\xi(PV_i)=PV_{jm}=\begin{cases}(V_{jm},L_{jm},V_{jmp}\cup V_i) & \text{if } L_{jm}=L_i+w_{ijm}\\ (V_{jm},L_i+w_{ijm},V_i) & \text{if } L_{jm}<L_i+w_{ijm}\\ (V_{jm},L_{jm},V_{jmp}) & \text{if } L_{jm}>L_i+w_{ijm}\end{cases}$$

如果$L_{jm}=L_i+w_{ijm}$，将V_i添加到前序节点集合V_{jmp}，并将节点V_{jm}入队；如果$L_{jm}<L_i+w_{ijm}$，取$L_{jm}=L_i+w_{ijm}$, $V_{jmp}=\{V_i\}$，并将节点V_{jm}入队；否则不变。

(4)重复第(3)步，直到队列为空。

编码图$CG(V_0)$生成后，从汇点PV_i开始进行反向搜索，其中，L_i为关键路径长度值，从V_{ip}找到前序节点，以此递归，直到源点V_0，求出从V_i到V_0的所有关键路径以及关键活动，算法 2 描述如下：

(1)创建 AOE 网的扩展邻接表表示法。

(2)根据算法 1，构造源点V_0编码图$CG(V_0)$。

(3)在编码图$CG(V_0)$中，如果节点PV_k的分量L_k非 0，则为源点V_0到节点V_k的关键路径的长度，从汇点出发，利用反向搜索，可求出所有关键路径及关键活动；否则无路径。

3.3 算法分析

(1)存储空间消耗分析

设$G=(V, E, W)$是一个具有n个节点的AOE网，现有算法采用相邻矩阵表示，需要开辟辅助数组$dist[n]$，其中，数组中每个元素包括 2 个字段：len字段表示源点V_0到其他节点的距离；per字段的值是从V_0到该节点前一节点的序号，因此，共需要n^2+2n个基本存储单元。在本文算法中，图G用扩展邻节表形式存储，编码图表头数组需要 $3n$个存储单元，边表需要 $3e$个(e是图G的边数)存储单元和 $2(n+m)$个前序节点存储单元(m为重复前序节点数)，因此，共需要 $3n+3e+2(n+m)$个基本存储单元，节省了存储空间。

(2)时间复杂度分析

在编码图的构造算法中，每个节点至多入队一次，因此，外循环次数为n。当节点出队后，内循环次数等于节点V_i的出度。由于访问所有节点的邻接点的总时间复杂度为

$$O(d_0+d_1+d_2+\cdots+d_{n-1})=O(e)$$

因此，构造编码图$CG(V_0)$的时间复杂度为$O(n+e)$，优于现有算法的时间复杂度$O(n^2)$。

4 算法应用举例

以图 4 为例，求解源点V_0到汇点V_8的所有关键路径及其长度。

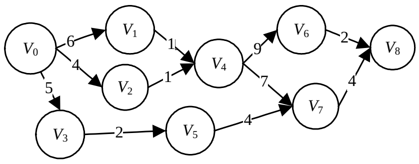


图 4 AOE网G_2

求解步骤如下：

(1)创建G_2的扩展邻接表表示法，如图 2 所示。其中，表头节点的指向前序节点指针域赋空(NULL)，源点到各节点最长路径值为 0。

(2)根据算法 1，构造节点V_0的编码图$CG_2(V_0)$，如图 3 所示，$CG_2(V_0)$的扩展邻接表表示法如图 5 所示。

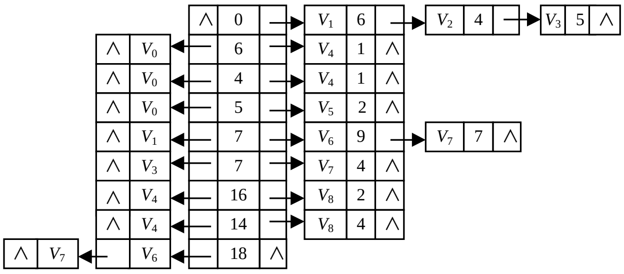


图 5 编码图$CG_2(V_0)$的扩展邻接表表示法

(3)在编码图$CG_2(V_0)$中，利用反向搜索，从汇点V_8开始直到源点V_0，求出V_0到V_8的所有关键路径及活动，求得AOE网的 2 条关键路径如图 6 所示。

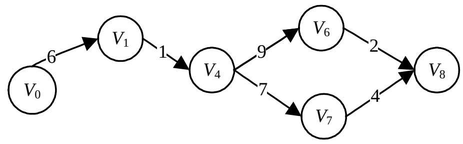


图 6 求得的 AOE 网的关键路径

5 结束语

本文通过定义源点编码图概念，将图的邻接表的存储结构进行扩充，提出了一种新的不需要拓扑排序的求解关键路径算法，与现有关键路径算法相比，其简单直观，而且存储空间消耗小，算法的时间复杂度降低了一个数量级，并能求出从源点到汇点的所有关键路径、关键活动和长度。事实上，可从源点编码图的任意节点出发，反向搜索直到源点，求出 AOE 网中任意节点到源点的所有关键路径、关键活动及长度(如果存在)。但如何扩充本算法，对有回路的有向图作出判断和处理，将是进一步研究的课题。

参考文献

[1] 严蔚敏, 吴伟民. 数据结构(C 语言版)[M]. 北京: 清华大学出版社, 1997.

[2] 徐凤生, 黄 倩. 关键路径求解新算法[J]. 计算机应用, 2004, 24(12): 108-109.

[3] 孟繁祯. 求解关键路径的一个算法[J]. 计算机工程, 2001, 21(4): 6-9.

[4] 刘 芳, 王 玲. 基于动态规划思想求解关键路径的算法[J]. 计算机应用, 2006, 26(6): 1440-1442.