



# Revivedsun的专栏

传播公益品牌，支持慈善事业  
为公益机构提供免费推广服务  
[open.baidu.com](http://open.baidu.com)

目录视图

摘要视图

RSS 订阅

评论送书 | 7月书讯：众多畅销书升级！ CSDN日报20170727——《想提高团队技术，来试试这个套路！》 评论送书 | 机器学习、Java虚拟机、微信开发

## 在有向无环图中求最长路径

2016-04-30 16:51 2831人阅读 评论(0)

分类：  
算法学习 ( 7 )

原文地址：<http://www.geeksforgeeks.org/find-longest-path-directed-acyclic-graph/>

给定一个带权有向无环图及源点S, 在图中找出从S出发到图中其它所有顶点的最长距离。

对于一般的图，求最长路径并不向最短路径那样容易，因为最长路径并没有最优子结构的属性。实际上求最长路径属于NP-Hard问题。然而，对于有向无环图，最长路径问题有线性时间的解。思路与通过使用拓扑排序在线性时间求最短路径[1]一样。

首先初始化到所有顶点的距离为负无穷大，到源点的距离为0，然后找出拓扑序。图的拓扑排序代表一个图的线性顺序。（图b是图a的一个线性表示）。  
当找到拓扑序后，逐个处理拓扑序中的所有顶点。对于每个被处理的顶点，通过使用当前顶点来更新到它的邻接点的距离。

个人资料



积分：1644  
等级：BLOG 4  
排名：千里之外

原创：58篇 转载：0篇  
译文：24篇 评论：6条

文章搜索

文章分类

算法学习 (8)

OJ练习 (18)

drools 5.X (1)

Java (13)

eclipse (1)

jquery (1)

MyBatis (4)

antlr (7)

jersey (4)

dubbo (9)

mysql (8)

rabbitmq (3)

spring相关 (3)

Apache-poi (2)

quartz (1)

kafka (1)

storm (1)

文章存档

2017年07月 (4)

2017年06月 (2)

2017年05月 (3)

2017年04月 (4)

2017年03月 (4)

展开

阅读排行

传播公益品牌，支持慈善事业

为公益机构提供免费推广服务

open.baidu.com

The Definitive Antlr 4 第7章...

(4299)

dubbo项目中使用logback输...

(3648)

MyBatis异常 Error setting dr...

(3638)

向无环图中求最长路径

(2824)

宜人贷

市场有风险 投资需谨慎

www.yirendai.com

借款5万

日息低至13元

凭信用卡申请

快至30分钟放款

Dubbo负载均衡：一致性Hash...

(0)

2178 表达式运算Cuties

(0)

Spring-boot实例学习之 Co...

(0)

1005 生日礼物

(0)

3130 CYD刷题

(0)

快速幂总结

(0)

2548 自然数积分解

(0)

推荐文章

\* CSDN日报20170725——《新的开始，从研究生到入职亚马逊》

\* 深入剖析基于并发AQS的重入锁(Reentrant Lock)及其Condition实现原理

\* Android版本的"Wannacry"文件加密病毒样本分析(附带锁机)

\* 工作与生活真的可以平衡吗？

\* 《Real-Time Rendering 3rd》提炼总结——高级着色：BRDF及相关技术

\* 《三体》读后思考-泰勒展开/维度打击/黑暗森林

最新评论

Dubbo负载均衡：一致性Hash的实现分析 yizishou ：不错

Spring-amqp 1.6.1 生产者与消费者消息... ytt9898 ：写的不错，原来消息数量大于消费者数量才会切换消费者

基于JDK动态代理实现的接口链式调用(Fl... FserSuN ：@qq\_21088015: 感谢指正。

基于JDK动态代理实现的接口链式调用(Fl... qq\_21088015 ：field.set(obj, value);field.setAccessible(true);顺序...

MyBatis异常 Error setting driver on Un... FserSuN ：@qq\_31238727:所引用的属性值所在的文件

MyBatis异常 Error setting driver on Un... 吴名霄 ：你好 属性文件是哪一个？

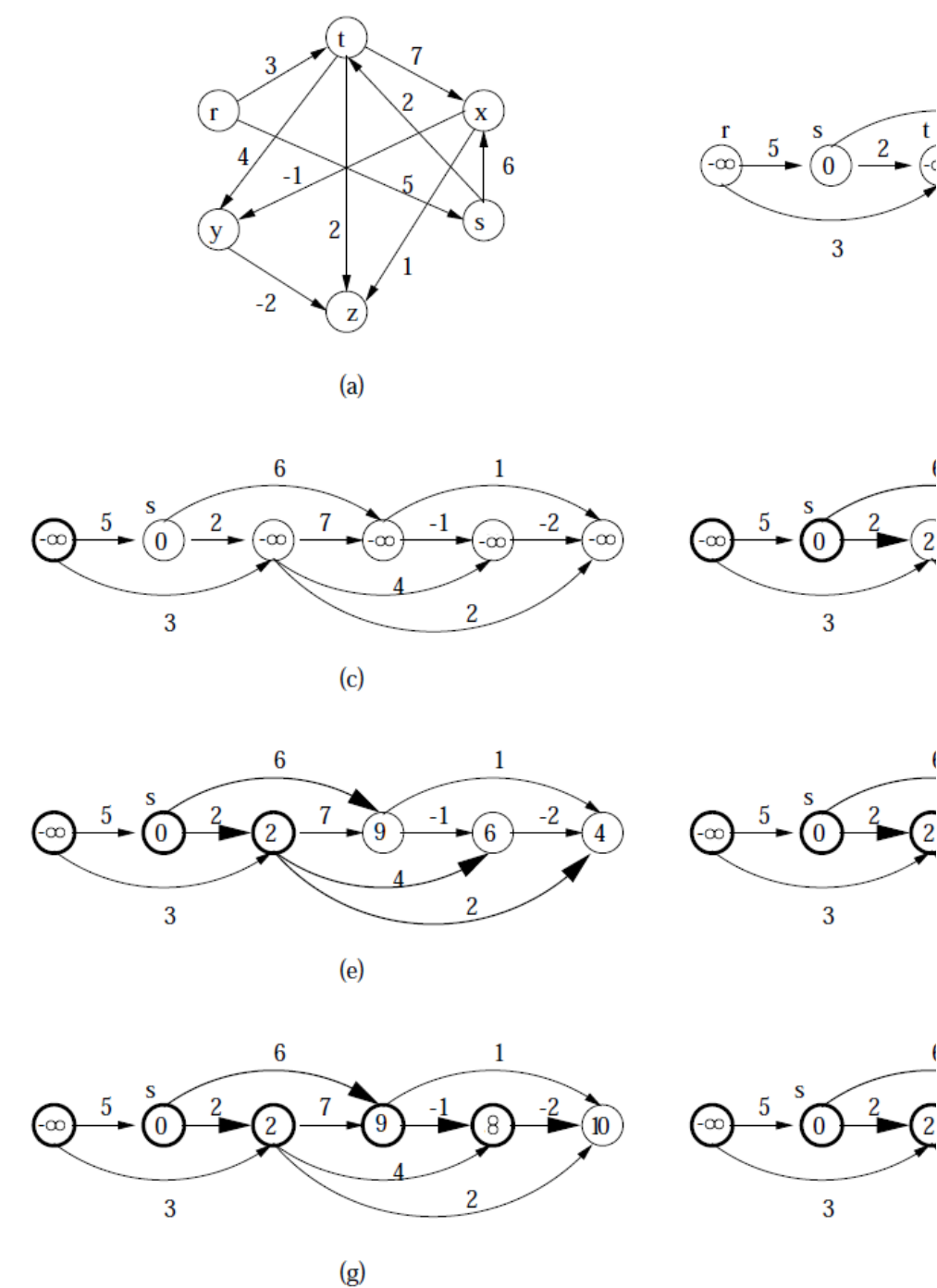


图 (b) 中，到点s的距离初始化为0, 到其它点的距离初始化为负无穷大，而图 (b) 中的边表示图 (a) 中边的权值。

图 (c) 中，求得从s到r的距离为负无穷。

图 (d) 中，求得s到t的最长距离为2, 到x的最长距离为6。

图 (e) 至图 (h) 依次求得可达点间的最长距离。

下面是寻找最长路径的算法

1. 初始化  $dist[] = \{NINF, NINF, \dots\}$  ,  $dist[s] = 0$  。s是源点，NINF表示负无穷。dist表示源点到其它点的最长距离。
2. 建立所有顶点的拓扑序列。
3. 对拓扑序列中的每个顶点u执行下面算法。

```
对u的每个邻接点v
if (dist[v] < dist[u] + weight(u, v)) .....dist[v] = dist[u] + weight(u, v)
```

下面是C++的实现。

传播公益品牌，支持慈善事业  
为公益机构提供免费推广服务  
[open.baidu.com](http://open.baidu.com)



[cpp]

```
01. // A C++ program to find single source longest distances in a DAG
02. #include <iostream>
03. #include <list>
04. #include <stack>
05. #include <limits.h>
06. #define NINF INT_MIN
07. using namespace std;
08.
09. //图通过邻接表来描述。邻接表中的每个顶点包含所连接的顶点的数据，以及边的权值。
10. class AdjListNode
11. {
12.     int v;
13.     int weight;
14. public:
15.     AdjListNode(int _v, int _w) { v = _v; weight = _w;}
16.     int getV() { return v; }
17.     int getWeight() { return weight; }
18. };
19.
20. // Class to represent a graph using adjacency list representation
21. class Graph
22. {
23.     int V; // No. of vertices'
24.
25.     // Pointer to an array containing adjacency lists
26.     list<AdjListNode> *adj;
27.
28.     // A function used by longestPath
29.     void topologicalSortUtil(int v, bool visited[], stack<int> &Stack);
30. public:
31.     Graph(int V); // Constructor
32.
33.     // function to add an edge to graph
34.     void addEdge(int u, int v, int weight);
35.
36.     // Finds longest distances from given source vertex
37.     void longestPath(int s);
38. };
39.
40. Graph::Graph(int V) // Constructor
41. {
42.     this->V = V;
43.     adj = new list<AdjListNode>[V];
44. }
45.
46. void Graph::addEdge(int u, int v, int weight)
47. {
48.     AdjListNode node(v, weight);
49.     adj[u].push_back(node); // Add v to u's list
50. }
51.
52. // 通过递归求出拓扑序列。详细描述，可参考下面的链接。
53. // http://www.geeksforgeeks.org/topological-sorting/
54. void Graph::topologicalSortUtil(int v, bool visited[], stack<int> &Stack)
55. {
56.     // 标记当前顶点为已访问
57.     visited[v] = true;
58.
59.     // 对所有邻接点执行递归调用
60.     list<AdjListNode>::iterator i;
61.     for (i = adj[v].begin(); i != adj[v].end(); ++i)
62.     {
63.         AdjListNode node = *i;
64.         if (!visited[node.getV()])
65.             topologicalSortUtil(node.getV(), visited, Stack);
66.     }
67.
68.     // 当某个点没有邻接点时，递归结束，将该点存入栈中。
69.     Stack.push(v);
70. }
```

传播公益品牌，支持慈善事业  
为公益机构提供免费推广服务  
[open.baidu.com](http://open.baidu.com)



[cpp]

```

01. // 根据传入的顶点，求出到其它点的最长路径。longestPath使用了
02. // topologicalSortUtil() 方法获得顶点的拓扑序。
03. void Graph::longestPath(int s)
04. {
05.     stack<int> Stack;
06.     int dist[V];
07.
08.     // 标记所有的顶点为未访问
09.     bool *visited = new bool[V];
10.     for (int i = 0; i < V; i++)
11.         visited[i] = false;
12.
13.     // 对每个顶点调用topologicalSortUtil，最终求出图的拓扑序列存入到Stack中。
14.     for (int i = 0; i < V; i++)
15.         if (visited[i] == false)
16.             topologicalSortUtil(i, visited, Stack);
17.
18.     // 初始化到所有顶点的距离为负无穷
19.     // 到源点的距离为0
20.     for (int i = 0; i < V; i++)
21.         dist[i] = NINF;
22.     dist[s] = 0;
23.
24.     // 处理拓扑序列中的点
25.     while (Stack.empty() == false)
26.     {
27.         // 取出拓扑序列中的第一个点
28.         int u = Stack.top();
29.         Stack.pop();
30.
31.         // 更新到所有邻接点的距离
32.         list<AdjListNode>::iterator i;
33.         if (dist[u] != NINF)
34.         {
35.             for (i = adj[u].begin(); i != adj[u].end(); ++i)
36.                 if (dist[i->getV()] < dist[u] + i->getWeight())
37.                     dist[i->getV()] = dist[u] + i->getWeight();
38.         }
39.     }
40.
41.     // 打印最长路径
42.     for (int i = 0; i < V; i++)
43.         (dist[i] == NINF)? cout << "INF " : cout << dist[i] << " ";
44. }

```

[cpp]

```

01. // Driver program to test above functions
02. int main()
03. {
04.     // Create a graph given in the above diagram. Here vertex numbers are
05.     // 0, 1, 2, 3, 4, 5 with following mappings:
06.     // 0=r, 1=s, 2=t, 3=x, 4=y, 5=z
07.     Graph g(6);
08.     g.addEdge(0, 1, 5);
09.     g.addEdge(0, 2, 3);
10.     g.addEdge(1, 3, 6);
11.     g.addEdge(1, 2, 2);
12.     g.addEdge(2, 4, 4);
13.     g.addEdge(2, 5, 2);
14.     g.addEdge(2, 3, 7);
15.     g.addEdge(3, 5, 1);
16.     g.addEdge(3, 4, -1);
17.     g.addEdge(4, 5, -2);
18.
19.     int s = 1;
20.     cout << "Following are longest distances from source vertex " << s << " \n";
21.     g.longestPath(s);
22.
23.     return 0;
24. }

```

输出结果:

传播公益品牌，支持慈善事业  
为公益机构提供免费推广服务  
[open.baidu.com](http://open.baidu.com)



```
[cpp]
01. 从源点1到其它顶点的最长距离
02. INF 0 2 9 8 10
```

**时间复杂度：**拓扑排序的时间复杂度是O(V+E).求出拓扑顺序后，对于每个顶点，通过循环找出所有邻接点，时间复杂度为O(E)。所以内部循环运行O(V+E)次。  
因此算法总的时间复杂度为O(V+E)。

[1] <http://www.geeksforgeeks.org/shortest-path-for-directed-acyclic-graphs/>

顶 0  
踩 0

- [上一篇](#) MyBatis异常 Error setting driver on UnpooledDataSource. Cause: java.lang.ClassNotFoundException:
- [下一篇](#) Java中的静态类以及嵌套类



猜你在找

- 【直播】机器学习&深度学习系统实战（唐宇迪）
- 【直播回放】深度学习基础与TensorFlow实践（王琛）
- 【直播】机器学习之凸优化（马博士）
- 【直播】机器学习之概率与统计推断（冒教授）
- 【直播】TensorFlow实战进阶（智亮）
- 【直播】Kaggle 神器：XGBoost 从基础到实战（冒教...
- 【直播】计算机视觉原理及实战（屈教授）
- 【直播】机器学习之矩阵（黄博士）
- 【直播】机器学习之数学基础
- 【直播】深度学习30天系统实训（唐宇迪）

查看评论

暂无评论

发表评论

用户名： THUChina  
评论内容：

提交

\* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

[公司简介](#) | [招贤纳士](#) | [广告服务](#) | [联系方式](#) | [版权声明](#) | [法律顾问](#) | [问题报告](#) | [合作伙伴](#) | [论坛反馈](#)

[网站客服](#)   [杂志客服](#)   [微博客服](#)   [webmaster@csdn.net](mailto:webmaster@csdn.net)   400-660-0108 | [北京创新乐知信息技术有限公司 版权所有](#) | [江苏知之为计算机有限公司](#) |

[传播公益品牌，支持慈善事业](#)  
为公益机构提供免费推广服务  
[open.baidu.com](http://open.baidu.com)   2017, CSDN.NET, All Rights Reserved   

