**目录**

[**§1 概括** 1](#_Toc392184912)

[**§2 Tological Sorting Solution** 2](#_Toc392184913)

[**§3 Application** 3](#_Toc392184914)

**参考文献：**

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

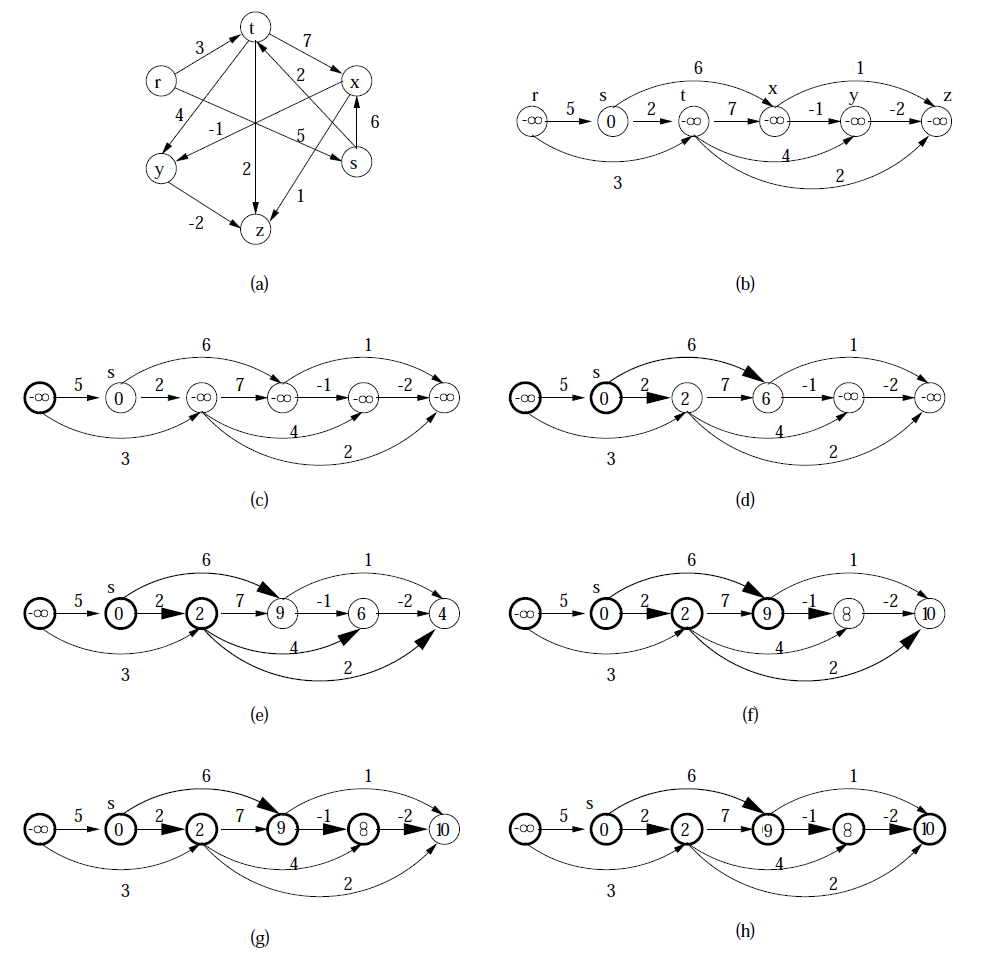
**§1 概括**

In [graph theory](http://en.wikipedia.org/wiki/Graph_theory) and [theoretical computer science](http://en.wikipedia.org/wiki/Theoretical_computer_science), the **longest path problem** is the problem of finding a [simple path](http://en.wikipedia.org/wiki/Path_(graph_theory)) of maximum length in a given graph. A path is called simple if it does not have any repeated vertices; the length of a path may either be measured by its number of edges, or (in [weighted graphs](http://en.wikipedia.org/wiki/Weighted_graph)) by the sum of the weights of its edges. In contrast to the [shortest path problem](http://en.wikipedia.org/wiki/Shortest_path_problem), which can be solved in polynomial time in graphs without negative-weight cycles, the longest path problem is [NP-hard](http://en.wikipedia.org/wiki/NP-hard), meaning that it cannot be solved in [polynomial time](http://en.wikipedia.org/wiki/Polynomial_time) for arbitrary graphs unless [P = NP](http://en.wikipedia.org/wiki/P_%3D_NP). Stronger hardness results are also known showing that it is difficult to [approximate](http://en.wikipedia.org/wiki/Approximation_algorithm). However, it has a [linear time](http://en.wikipedia.org/wiki/Linear_time) solution for [directed acyclic graphs](http://en.wikipedia.org/wiki/Directed_acyclic_graph), which has important applications in finding the [critical path](http://en.wikipedia.org/wiki/Critical_path_method) in scheduling problems. The idea is similar to [linear time solution for shortest path in a directed acyclic graph](http://www.geeksforgeeks.org/shortest-path-for-directed-acyclic-graphs/), we use [Tological Sorting](http://www.geeksforgeeks.org/topological-sorting/" \t "_blank).

**§2** [**Tological Sorting**](http://www.geeksforgeeks.org/topological-sorting/) **Solution**

We initialize distances to all vertices as minus infinite and distance to source as 0, then we find a[topological sorting](http://www.geeksforgeeks.org/topological-sorting/) of the graph. Topological Sorting of a graph represents a linear ordering of the graph (See below, figure (b) is a linear representation of figure (a) ). Once we have topological order (or linear representation), we one by one process all vertices in topological order. For every vertex being processed, we update distances of its adjacent using distance of current vertex.

Following figure shows step by step process of finding longest paths.

[](http://d2o58evtke57tz.cloudfront.net/wp-content/uploads/LongestPath.png)

Following is complete algorithm for finding longest distances.  
**1)** Initialize dist[] = {NINF, NINF, ….} and dist[s] = 0 where s is the source vertex. Here NINF means negative infinite.  
**2)** Create a toplogical order of all vertices.  
**3)** Do following for every vertex u in topological order.  
………..Do following for every adjacent vertex v of u  
………………if (dist[v] < dist[u] + weight(u, v))  
………………………dist[v] = dist[u] + weight(u, v)

**§3 Application**

The [critical path method](http://en.wikipedia.org/wiki/Critical_path_method) for scheduling a set of activities involves the construction of a directed acyclic graph in which the vertices represent project milestones and the edges represent activities that must be performed after one milestone and before another; each edge is weighted by an estimate of the amount of time the corresponding activity will take to complete. In such a graph, the longest path from the first milestone to the last one is the critical path, which describes the total time for completing the project.[[4]](http://en.wikipedia.org/wiki/Longest_path_problem#cite_note-sedge-4)

Longest paths of directed acyclic graphs may also be applied in [layered graph drawing](http://en.wikipedia.org/wiki/Layered_graph_drawing): assigning each vertex *v* of a directed acyclic graph *G* to the layer whose number is the length of the longest path ending at *v* results in a layer assignment for *G* with the minimum possible number of layers.