ePAL - Minimum Spanning Tree

Radek Mařík
Marko Genyk-Berezovskyj

ČVUT FEL, K13133

October 3, 2012

Outline

1 Fundamentals

Outline

1 Fundamentals

General Assumption in the Following Examples

Unless stated explicitly, we assume simple connected weighted graphs without loops.

Example 1

We add the same nonzero constant c to the price of each edge in a graph.

In what relationship will the minimal skeleton of original and modified graphs be ?

Example 2

We multiply all edge weights with the same nonzero constant c.

In what relationship will the minimal skeleton of original and modified graphs be ?

Example 3

On your computer, you have to find the minimum spanning tree of a weighted complete graph. Estimate what is the order of maximum number of nodes such a graph can have to solve the task through the night until the next day.

Example 4

A greedy robber takes as most expensive as possible edges from the graph (nodes are left in place). But he must not disconnect the graph into two or more components, because he would be immediately detected and apprehended.

When you remove the maximum of what can be done, will the graph's minimum spanning tree be left?

Example 5

Is it enough to satisfy that all edges have a different cost to a minimum spanning tree is determined uniquely?

Example 6

We need to find a spanning tree of the graph, not necessarily minimal, with the requirement that the price of each edge of the searched spanning tree must be in the interval $< c1, c2 >$.

Is it necessary to use an algorithm for finding minimum spanning trees, or just a simple procedure?

Example 7

Kruskal's algorithm described in the lecture prescribes that we first sort all the edges using their prices. It can be expected that the edges of a high price are not part of the minimum spanning tree, and therefore they are sorted entirely unnecessarily. We propose a modified algorithm that only inserts edges into the priority queue at the beginning and then the next cheapest edge of the queue is always selected during the construction of the minimum spanning tree. We want to know which option is preferable.

1. Will the asymptotic complexity of the algorithm be changed?
2. Will the difference between the original and modified algorithm be clearer when applied to a graph with many or with few edges?

Example 8

Remind yourself how many edges can a planar graph with *n* nodes have and then answer the question: What algorithm is preferably used to find a minimum spanning tree of the graph and what will its complexity be depending only on *n*?

Example 9

How is a maximum spanning tree searched for using an algorithm for finding minimum spanning tree?

Example 10

Graphland headquarters search for the cheapest minimum spanning tree of a graph with the restriction that some edges of the graph are already selected, they must be in the result tree regardless of the price. Therefore, one can expect that the result tree will not be the cheapest possible. Nevertheless, in this case, you can use one of the minimum spanning tree search algorithm to fulfill the task. Fortunately, no prescribed subset of edges forms a cycle.

Example 11

It happens in the previous task some prescribed edges form together a cycle in the graph. Therefore, the headquarters modify the problem so that they do not require a minimum spanning tree but the cheapest interconnection of all graph nodes, and insist that all pre-specified edges must be used. Show that there is no difference in a solution regarding to the previous task.

Example 12

The next task is very easy, once you "get through" its specification:
A feedback set of edges of the graph is each such a set of edges that contains at least one edge of each cycle of the graph. It follows that if the feedback set is removed from the graph, a graph with no cycle is left. How do we determine what is the cheapest feedback set? (Price of the set of edges is the sum of the prices of all edges in the set.)

Example 13

Graphs $G1$ and $G2$ have an identical structure (they are isomorphic), but there are nearly all edge with different price in $G1$, while almost all edges of $G2$ have the same cost.

What kind of a minimum spanning tree search algorithm should be used for $G1$ and what for $G2$ and why?

Example 14

We found the minimum spanning tree of a graph supplied with many millions of vertices and transferred it to the customer. In the afternoon the customer calls that the graph specification contains an error and that in fact the edge between vertices 2075154 and 11439446 is about 17% cheaper than as described in the original specification. All graph data and the minimum spanning tree are still on our disk. We should determine in linear time with regard to the number of vertices if this change affects the shape and the price of the minimum spanning tree, and if so, to give the shortest possible fix that can convey back using the phone.

Example 15

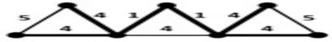
Repeat the analysis of the previous task for the case that the incorrect edge entered originally will happen more expensive in fact.

Example 16

Assume that the graph is specified as a matrix of weights of individual edges. An outstanding value in this matrix (e.g. infinity, minimum / maximum value of a numeric type, *NaN*, etc.) indicates that an edge between the vertices does not exist. Modify Jarnik-Prim's algorithm so that it does not depend on the number of edges in the graph and has complexity $\Theta(n^2)$, where n is the number of nodes in the graph.

Example 17

Obviously, the minimum spanning tree can be unique in a single graph, even if some of the edges are equal. For example, as shown in this figure.



How do we know if the minimum spanning tree in the graph is only one? At first we thought: When you add all edges having the same cost c into the minimum spanning in the Kruskal algorithm then the minimum spanning tree is unique. But it does not apply in the picture, the middle edge with a value of 4 is not added to the minimum spanning tree although the minimum spanning is just one. Therefore, we propose a finer criterion: when all the edges that have the same price c are added into the minimum spanning tree in Kruskal's algorithm and each such edge links nodes in different subtrees built till the edges with cost c encountered then the minimum spanning tree exists just one. Now we have two questions. Is it valid really? If yes, what is the complexity of the verification criteria?
The criterion should be treated carefully because at the end of construction of minimum spanning trees one might want even to add unnecessarily too many edges of the same price, but it has no major impact, as you can see.

Example 18

The difference between the most expensive and the cheapest edge of graph $G = (V, E)$ is called a dispersion value of the minimum spanning tree. Find the minimum spanning tree with minimum dispersion in time $O(|E|2log(|V|))$. Try to use a list of edges ordered according to their weights.

References I