Tutorial 4: Spanning tree
Artificial Intelligence

G.Guérard

SOME SOLUTIONS MAY HAVE FALSE CLUES – SO BE CAREFUL

Spanning tree

Exercise 1

There are 5 cities. The cost of building a road directly between i and j is the entry $a_{i,j}$ in the matrix below. An indefinite entry indicates that the road cannot be built. Determine the least cost of making all the cities reachable from each other.

$$\begin{pmatrix} 0 & 3 & 5 & 11 & 9 \\ 3 & 0 & 3 & 9 & 8 \\ 5 & 3 & 0 & \infty & 10 \\ 11 & 9 & \infty & 0 & 7 \\ 9 & 8 & 10 & 7 & 0 \end{pmatrix}$$

Solution

We order the edges according to the weights: 12, 23, 13, 45, 25, 15, 24, 35, 14 (raw-column). Kruskal's Algorithm accepts edges 12, 23, then rejects 13, then accepts 45, 25, and then it stops. Thus, the least cost to build the road network is 3 + 3 + 7 + 8 = 21.

Exercise 2

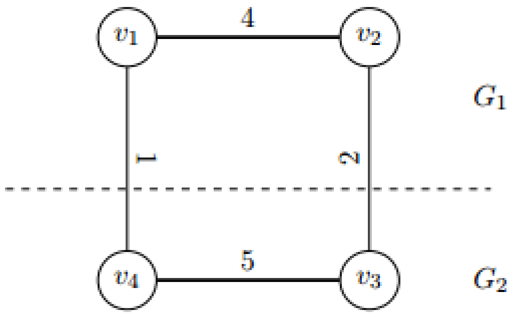
Professor Herr Guerard proposes a new divide-and-conquer algorithm for computing minimum spanning trees, which goes as follows.

Given a graph G = (V, E), partition the set V of vertices into two sets V_1 and V_2 such that $|V_1|$ and $|V_2|$ differ by at most 1. Let E1 be the set of edges that are incident only on vertices in V_1, and let E_2 be the set of edges that are incident only on vertices in V_2. Recursively solve a minimum-spanning-tree problem on each of the two subgraphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. Finally, select the minimum-weight edge in E that crosses the cut V1, V2, and use this edge to unite the resulting two minimum spanning trees into a single spanning tree.

Either argues that the algorithm correctly computes a minimum spanning tree of G, or provide an example for which the algorithm fails. Found an example where it works and where it doesn't work.

Solution

We claim that the algorithm will fail. A simple counter example is shown below. Graph G = (V, E) has four vertices: {v_1, v_2, v_3, v_4}, and is partitioned into subsets G_1 with V_1 = {v_1, v_2} and G_2 with V_2 = {v_3, v_4}. The MST of G_1 has weight 4, and the MST of G_2 has weight 5, and the minimum-weight edge crossing the cut (V_1, V_2) has weight 1, in sum the spanning tree forming by the proposed algorithm follows {v_2, v_1, v_4, v_3} which has weight 10. On the contrary, it is obvious that the MST of G follows {v_4, v_1, v_2, v_3} with weight 7. Hence the proposed algorithm fails to obtain an MST.



(Take another example than the solution for the oral test!)

Exercise 3

Show that if G is a weighted graph and e is an edge whose weight is smaller than that of any other edge, then e must belong to every minimum weight spanning tree for G.

Solution

Suppose that T is a minimum weight spanning tree for G that does not contain the edge e. Then Consider the graph T+e. This graph must contain a cycle C that contains the edge e. Let f be an edge of C different from e, and set T*=T+e−f. Then T* is also a spanning tree for G, but $w(T^*)=w(T+e-f)=w(T)+w(e)-w(f) < w(T)$, contrary to T being a minimum weight spanning tree. Hence no such tree T (i.e., without e) can exist.

Exercise 4

Show that if all the weights of the weighted graph G are distinct, then there is a unique minimum weight spanning tree for G.

Solution

The proof somewhat mimics that of the proof of Kruskal's Algorithm. Suppose that T is a tree generated by Kruskal's Algorithm (in fact, a moment's thought shows that with the conditions of the problem, only one such tree could be generated). We claim there is no other minimum weight spanning trees for G. Suppose (and we will show this leads to a contradiction) that there are other minimum weight spanning trees, and choose one, T'. Then suppose that e is the first edge of T that is not in T'. In other words, suppose that the edges of T, in the order they were added to form T, are e_1, e_2 , …, e_k , …e_{n-1} and that $e = e_k$ and for all i<k, $e_i \in T'$. Let C be the cycle in T'+ e that contains e. let f

be an edge of C that is not in T'. We note that by the nature of Kruskal's algorithm, the weight of f must be greater than the weight of e. This is because at the time we placed e into T, f was also available and would not have produced a cycle (since all the edges of T up to that point are in T' as well). So if w(f)<w(e), we would have used f at that juncture. So now set T*=T'+e−f is a spanning tree of weight less than T', a contradiction.

Exercise 5

Consider a "reversed" Kruskal's algorithm for computing a MST. Initialize T to be the set of all edges in the graph. Now, consider edges from largest to smallest cost. For each edge, delete it from T if that edge belongs to a cycle in T. Assuming all the edge costs are distinct, does this new algorithm correctly compute a MST?

Solution

Yes, it does. At stage k (starting a k=1), the algorithm considers the k-th largest cost edge. If that edge belongs to a cycle in the remaining graph T, then all edges in that cycle (and indeed in T) must have smaller cost than the edge being considered. Thus, the edge cannot belong to the MST (by the previous question). The algorithm cannot terminate with T having a cycle, since the algorithm would have considered each edge in such a cycle and would have removed the edge of the largest cost when it considered that edge. The algorithm also cannot terminate with T being disconnected, since edges are only removed when they belong to a cycle, and disconnecting an edge that belongs to a cycle does not disconnect the graph. Thus the algorithm terminates with T being a spanning tree. It is the MST because all the edges that were removed have the property that they cannot belong to the MST. Since the only edges that could belong to the MST are the ones that remain, and they indeed define a spanning tree, it must be the MST.