MATH.APP.270 Algorithms for graphs

Programming assignment 3 2022

There are several ways to compute a minimal spanning tree (MST). Prim's algoritm, given in the course notes, is just one alternative. In this assignment you will write a procedure for another algorithm.

Let $G = (V, E, w)$ be a connected weighted graph. The following procedure, MSTCYCLE, takes such a graph as input and produces an MST.

```
1  procedure MSTCYCLE
2  input (V, E, w)
3  Ê = E
4  while there are still cycles in Ê do
5      find a cycle c in Ê
6      find the edge (x, y) in c whose weight is the largest
7      Ê := Ê \ {(x, y)}
8  end while
```

Your task in this assignment will be to write code for algorithm MSTCYCLE. Clearly many details are missing from the pseudocode. We next offer suggestions.

In line 5 of MSTCYCLE the task is to find one cycle in the remaining graph $(V, \hat{E}, w)$. A depth-first search (DFS) algorithm can be used to detect cycles. Naturally, one could run a DFS on the entire input graph and then simply use one of the cycles detected. However, it makes more sense to run the DFS only until the first cycle is detected. To do this one needs to make changes to the DFS algorithm presented in the course notes. The pseudocode in the following box contains a modified DFS which is intended to stop executing as soon as one cycle is found. With regard to the code in this box we note the following:

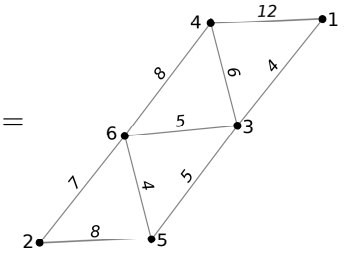
- DFSCYCLE is only called once in the 'main' procedure at line 7 since we are assuming that the input graph is connected.
- When a cycle is detected at line 22, procedure DFSCYCLE is terminated and an edge e is returned at line 23.
- Unlike the depth first search algorithm of the course notes, no finishing time $f[u]$ is included. In fact, it would probably be possible to remove the discovery time $d[u]$ as well.

After a cycle edge e is detected at line 5, the remainder of the cycle c can be found using the parent function $p[u]$. It is then a simple task to find which cycle edge has the largest weight at line 6. Finally this 'heaviest edge' must be removed at line 7. The **while**-loop continues until no more cycles exist.

```
1   input: weighted connected graph (V, E, w)
2   output: edge e, parent p[u], u ∈ V, discovery time d[u], u ∈ V
3   for u ∈ V do
4       col[u] := white, p[v] := nil
5   end for
6   t := 0
7   select some vertex u as the starting vertex
8   e = DFSCYCLE(u)
9
10  procedure DFSCYCLE(u)
11    t := t + 1, d[u] := t, col[u] := gray
12    for v ∈ {v | (u, v) ∈ E} do
13        if col[v] = white then
14            p[v] := u
15            e = DFSVISIT(v)
16            if e is not empty then
17                return e
18            end if
19        else
20            if col[v] = gray ∧ p[u] ≠ v then
21                edge (u, v) is part of a cycle
22                return (u, v)
23            end if
24        end if
25    end for
26    col[u] := black
27    return ()
28  end procedure
```

We will present an example of how the algorithm MSTCYCLE works. In this example we will use graph G_1 depicted next.

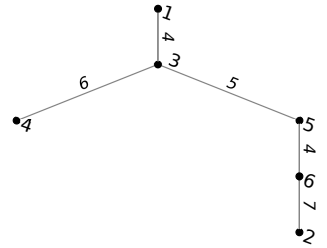
$G_1 =$



The following table contains one possible set of cycles detected and edges removed in each iteration of the **while**-loop of MSTCYCLE:

while-loop iteration	cycle c	edge removed
1	$\langle 4, 3, 1, 4\rangle$	$(1, 4)$
2	$\langle 4, 6, 2, 5, 3, 4\rangle$	$(4, 6)$
3	$\langle 6, 3, 5, 2, 6\rangle$	$(2, 5)$
4	$\langle 6, 3, 5, 6\rangle$	$(3, 6)$

Based on this table, the MST produced by MSTCYCLE would be the following:

$T_1 =$ 

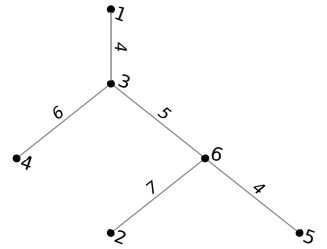
With regards to using DFSCYCLE for computing MSTs two remarks are necessary.

- The cycles detected are not unique, nor is the order in which they are detected. Both of these depend on the order of discovery of the vertices in DFSCYCLE.
- A graph may have several MSTs.

We demonstrate these two remarks, by again using graph G_1 as input to MSTCYCLE. It is possible that MSTCYCLE would produce the following results:

while-loop iteration	cycle c	edge removed
1	$\langle 4, 3, 1, 4 \rangle$	$(1, 4)$
2	$\langle 3, 5, 6, 4, 3 \rangle$	$(4, 6)$
3	$\langle 5, 3, 6, 5 \rangle$	$(3, 5)$
4	$\langle 5, 6, 2, 5 \rangle$	$(2, 6)$

From these results, the MST we obtain is the following:

$T_2 =$ 

Both T_1 and T_2 are correctly computed.

Your task in this assignment is to write a method or function MSTCYCLE. This function should meet the following specifications:

- The input to MSTCYCLE is a weighted connected graph.
- Two different items should be output by MSTCYCLE: (i) a list of the cycles detected in the order they are detected and (ii) the final MST.

Each cycle should be stored as a list. The final MST should be stored as a weighted graph.

Data for testing
A set of graphs for testing purposes will be published separately.