University of Washington
Department of Computer Science and Engineering
CSE 417, Winter 2020
Yiliang Wang

January 29, 2020

Homework 3 Problem 2

Problem 2 (10 points):

Consider a directed graph on n vertices, where each vertex has exactly one outgoing edge. This graph consists of a collection of cycles as well as additional vertices that have paths to the cycles, which we call the branches. Describe a linear time algorithm that identifies all of the cycles and computes the length of each cycle. You can assume that the input is given as an array A, where $A[i]$ is the neighbor of i, so that the graph has the edge $\langle i, A[i] \rangle$.

Answer:

```
def count_cycle(A):
    cur_cycle = 0
    cycle_label = [0 for i in range(len(A))]
    cycles = []

    for node in range(len(A)):
        if cycle_label[node] == 0:
            cur_cycle += 1
            cur = node
            while cycle_label[cur] == 0:
                cycle_label[cur] = cur_cycle
                cur = A[cur]
            if cycle_label[cur] == cur_cycle:
                cycles.append([])
                cycles[-1].append(cur)
                nxt = A[cur]
                while nxt != cur:
                    cycles[-1].append(nxt)
                    nxt = A[nxt]
    return cycles
```

proof:
Since each vertex has exactly one outgoing edge, any strongly connected components in this graph would not contain more than one cycle. Because to form two cycle within a strongly connected components, two cycle need to share at least one edge or at least one node. If one edge $u - v$ is shared, the other path between u and v should in contrary direction to form a closed cycles (from

v to u), this makes one node has two incoming edge and one node has two outgoing edges, which confict with single outgoing edge assumption. If one node is shared between two cycle, it must has two pairs of incoming and outgoing edge from both cycles. Hence each strongly connected components can only has one cycle and possibly branches to cycles. This garantee the algorithm would always halt when it found visited node with on same cycle, because traversal start from any node can and can only lead to a single cycle.