DFS Approach Analysis

1. canonical_cycle(cycle[0...n-1])

- Basic operation: comparison

$C(n) = \sum_{i=1}^{n} n = n^2$

$\in \Theta(n^2)$

2. checkCycles()

- Basic operation: Comparison

$C(n) = \sum_{i=1}^{n} (n-1) = \frac{(n-1)(n)}{2}$

Worst case: $\frac{(n-1)n}{2}$ asymptotic classification: $\in \Theta(n^2)$

3. dfsHelper(node, graph, visited[0..n-1], locations)

- Basic operation: Comparison

Let a_i = # of simple paths of length i from the intial vertex.

$T(n) = \sum_{i=1}^{n-1} i \cdot a_i$ so in worst case $a_i \leq n^{i-1}$

$= \sum_{i=1}^{n-1} i(n^{i-1}) = \sum_{i=1}^{n-1} n(n^{i-1}) = n \sum_{i=1}^{n-1} n^{i-1}$

$= n\left(\frac{n^{n-1}-1}{n-1}\right)$ $\in \Theta(n^n)$ ∴ Worst case time complexity is exponential

- Best case of input size n is when node pointer is null or the adjacency list for the starting vertex is empty

$T_{Best}(n) \in \Theta(1)$ is constant time

4. dfsApproach(graph)

- Basic operation: processing one adjacency list entry during DFS Traversal
- For adjacency list representation DFS runs in $\Theta(|V| + |E|)$
- For the best case for n nodes in the graph, each loop executes n times

$$T(n) = \sum_{i=1}^{n} C = C \sum_{i=1}^{n} 1 = n$$

$$\in \Theta(n)$$

best case is linear