Homework 8

Kevin Orr

June 25, 2017

1. The inner-most for loop runs $deg(V)$ times in the worst case, with each iteration taking O(1) time, and the call to `getNeighbors` taking $\mathrm{O}(deg(v_j))$ time. The outer for loop also runs $deg(v)$ times, with at most O(1) overhead per iteration in addition to the inner for loop, and the call to `getNeighbors` taking $\mathrm{O}(deg(v_i))$ time.

$$\begin{aligned}
T_{for}(v_i, v_j) &= \mathrm{O(deg(v_i))} + \sum_{\mathrm{v_x \in N(V_i)}} \left[\mathrm{O(deg(v_j))} + \sum_{\mathrm{v_y \in N(v_j)}} \mathrm{O(1)} \right] \\
&= \mathrm{O(deg(v_i))} + \sum_{\mathrm{v_x \in N(V_i)}} [\mathrm{O(deg(v_j))} + \mathrm{O(deg(v_j))}] \\
&= \mathrm{O(deg(v_i))} + \sum_{\mathrm{v_x \in N(V_i)}} \mathrm{O(deg(v_j))} \\
&= \mathrm{O(deg(v_i))} + \mathrm{O(deg(v_i))O(deg(v_j))} \\
&= \mathrm{O(deg(v_i) + deg(v_i)deg(v_j))} \\
&= \mathrm{O(deg(v_i)deg(v_j))}
\end{aligned}$$

2.

$$
\begin{aligned}
T(m,n) &= \sum_{v_i \in V} \sum_{v_j \in V} T_{for}(v_i, v_j) \\
&= \sum_{v_i \in V} \sum_{v_j \in V} \mathrm{O(deg(v_i)deg(v_j))} \\
&= \sum_{v_i \in V} [\mathrm{O(deg(v_i)deg(v_1)) + O(deg(v_i)deg(v_2))} + \ldots + \mathrm{O(deg(v_i)deg(v_n))}] \\
&= \sum_{v_i \in V} [\mathrm{O(deg(v_i)deg(v_1) + deg(v_i)deg(v_2)} + \ldots + \mathrm{deg(v_i)deg(v_n))}] \\
&= \sum_{v_i \in V} [deg(v_i)\mathrm{O(deg(v_1) + deg(v_2)} + \ldots + \mathrm{deg(v_n))}] \\
&= \sum_{v_i \in V} [deg(v_i)\mathrm{O(2mn)}] \\
&= \mathrm{O(2mn)} \sum_{\mathrm{v_i \in V}} \mathrm{deg(v_i)} \\
&= \mathrm{O(2mn)O(2mn)} \\
&= \mathrm{O(4m^2n^2)} \\
&= \mathrm{O(m^2n^2)}
\end{aligned}
$$

3. This is the same time complexity as 2. The function will iterate over each pair of vertices no less than once if the graph is fully connected, and no more than once since it will skip over already-visited pairs. So it will iterate exactly n^2 times, making the total time complexity $\mathrm{O(m^2n^2)}$.