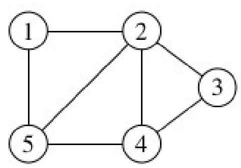
Algorithms Assignment 2

Jeremy Kwok 100341977

Part A

1. The eccentricity of a given vertex v is the greatest distance away from v to any other vertex in the graph. That is, if you calculate the shortest path between v and every other node in graph g, the eccentricity is the maximum shortest path between v and any of [v0,v1,v2....vn-1]

For Example, given this graph from class (1.2 from search slide):



Looking at the eccentricity of vertex 3:

3 is 2 away from 1
3 is 1 away from 2
3 is 1 away from 4
3 is 2 away from 5

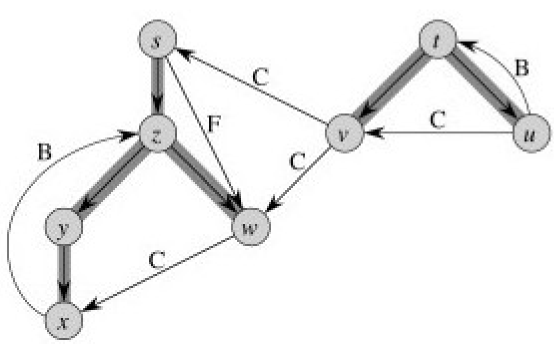
So the eccentricity of 3 is the maximum of these distances, so 2.

2. The radius of a given graph g is the minimum eccentricity of all of g's vertexes

3. The diameter of a given graph g is the maximum eccentricity of all of g's vertexes.

Example for 2 and 3:

Given this graph from class: (3.6 from search slide)



Eccentricity of s is 3 (s to x)
Eccentricity of t is 6 (t to x)
Eccentricity of u is 6 (u to x)
Eccentricity of v is 4 (v to x)

Eccentricity of w is 3 (w to y)
Eccentricity of x is 2 (x to y)
Eccentricity of y is 3 (y to w)
Eccentricity of z is 2 (z to x)

Therefore the minimum eccentricity and radius is 2, and the maximum eccentricity and diameter is 6.

4.

To compute eccentricity of a graph, we use a similar algorithm to BFS. We will calculate the distance from each node to each other node via BFS (eccentricity from vertex v to each other node, and then take the minimum and maximum of those calculated vertices.

Here we use a slightly modified BFS algorithm that also returns the list of distances d[] that it uses in the search.

```
def radius_diameter(Graph):
	V = V(Graph)
	for source in V:
		eccentricities = eccentricity(source,Graph)
	return min(eccentricities),max(eccentricities)

def eccentricity(source,Graph):
	V = V(Graph)
	for destination in V:
		if source != destination:
			distances.add(BFS(Graph, source))

	return distances
```

5.

The computational complexity of BFS is O(V+E).
The eccentricity function calls BFS for each vertex in the graph. Therefore, its computational complexity is BFS * V = O(V*(V+E)).
The radius_diameter function calls eccentricity for each vertex in the graph. Therefore, its computational complexity is eccentricity * V = O(V^2*(V+E))

Part B

(Note: we have emailed Ken prior, and asked if an adjacency matrix was mandatory, as using adjacency lists alone would be sufficient, and he told us this was acceptable).

3.
Output from the program:
Components and their Sizes:
0: 148
1: 244
2: 10
3: 16
4: 2
5: 2
6: 5
7: 3
8: 3
9: 5
10: 2
11: 6
12: 2
13: 2
14: 4
15: 3
16: 2
17: 2

Thus there are 18 different connected components, with sizes ranging from 244 professors as the largest component and many small components of 2 professors.

Each connected component in theory should represent a Department or Faculty at UOIT. Computer Science professors will teach the same courses as other Computer Science professors, and those professors would not share any courses with professors in other departments. In reality, there are some courses taught by professors of different faculties (for example a course may be taught by a Computer Science professor one year and an Engineering professor another) causing a small amount of large connected components.

4.

Output from the program:

Components and their Sizes:
0: 148
1: 244
2: 10
3: 16
4: 2
5: 2
6: 5
7: 3
8: 3
9: 5
10: 2
11: 6
12: 2
13: 2
14: 4
15: 3
16: 2
17: 2

Components and their Radii:
Radii of component 0: 10
Radii of component 1: 11
Radii of component 2: 3
Radii of component 3: 3
Radii of component 4: 1
Radii of component 5: 1
Radii of component 6: 2
Radii of component 7: 1
Radii of component 8: 1
Radii of component 9: 2
Radii of component 10: 1
Radii of component 11: 2
Radii of component 12: 1
Radii of component 13: 1
Radii of component 14: 1
Radii of component 15: 1
Radii of component 16: 1
Radii of component 17: 1

It is not surprising to note that the larger a component is, the higher its radius will be. This is true in our case since a single professor is only connected to a handful of other professors. In a highly connected graph, we could expect large components to still have small radii.