Graphs 2: Link Analysis

300958 Social Web Analysis

Week 10 Lab Solutions

- Write the function `normalise` that takes a vector `x` and returns the vector `x` divided by its sum.

```
normalise = function(x) {
  return(x/sum(x))
}
```

- Write the function `difference`, that returns the Euclidean distance between two vectors.

```
difference = function(x, y) {
    return(sqrt(sum((x - y)^2)))
}
```

- Check if the computed stationary distribution satisfies the stationary distribution equation.

```
  p - (T %*% p)
```

- Examine `p` to make sure it is the stationary distribution.

```
  p - (T %*% p)
```

- Create a new graph `g2` and compute its stationary distribution using the Power Method and the Eigenvalue decomposition.

```
g2 = graph.formula(1-2, 2-3, 2-4, 3-4, 1-5, 4-5)
A = get.adjacency(g2)
T = adjacency.to.probability(A)
p = stationary.distribution(T)
print(p)
e = eigen(T)
p = normalise(e$vectors[, 1])
print(p)
```

- Compute the stationary distribution of `gd1`

```
A = get.adjacency(gd1)
T = adjacency.to.probability(A)
e = eigen(T)
p = normalise(e$vectors[,1])
print(p)
```

- For each of the following graphs, plot them, identify why they are non-ergodic, then try to compute the stationary distribution. Take note of the eigenvalues of each transition probability matrix.

```
gd2 = graph.formula(1-+2, 2-+3, 4-+3, 3-+1)
plot(gd2)
A = t(as.matrix(get.adjacency(gd2)))
T = adjacency.to.probability(A)
e = eigen(T)
print(e$values)

gd3 = graph.formula(1+-2, 2-+3, 2-+4, 4-+3, 3-+1)
plot(gd3)
A = t(as.matrix(get.adjacency(gd3)))
T = adjacency.to.probability(A)
e = eigen(T)
print(e$values)

gd4 = graph.formula(1-+2, 2-+3, 3-+1, 4+-+5)
plot(gd4)
A = t(as.matrix(get.adjacency(gd4)))
T = adjacency.to.probability(A)
e = eigen(T)
print(e$values)
```

- Compute the PageRank for `gd3` and `gd4`.

```
A = t(as.matrix(get.adjacency(gd3)))
T = adjacency.to.probability(A)
M = alpha*T + (1-alpha)*J
p = stationary.distribution(M)
print(p)

A = t(as.matrix(get.adjacency(gd4)))
T = adjacency.to.probability(A)
M = alpha*T + (1-alpha)*J
p = stationary.distribution(M)
print(p)
```