Data Mining Homework 11

Madis Nõmme

May 14th, 2014

Task 1

- number of nodes in the network: *85591*
- number of edges in the network: *156217*
- number of nodes with a self-loop: *869*
- number of mutual connections or reciprocated edges: *35290*
- number of nodes with zero indegree (those that have only outgoing edges):
- number of nodes with zero outdegree (those that have only ingoing edges): 0
- degree distribution of the given network
- multiple values, in nodes, out nodes
- optionally calculate whatever measure you deem appropriate for better understanding
- starting with disconnected nodes will stop the virus from spreading

Task 2

B) - whats the property - SCC IN component affects all the SCC & out components - SCC OFF component always gets affected if virus comes in

proportions: IN 3/17 STRONG 7/17 OUT 4/17

```
DISCONNECTED 100-rest

Probability of 'large scale' epidemic:

  0.17*(0.41)+0.24 = 0.24+(0.17*0.41) = 0.309
```

Actually: you add up the IN & STRONG because these cause the OUT to get infected. Real answer: 7/17 + 3/17 = 10/17

Task 3

Differences: some models describe better some real world graphs.

Task 4

Real world: 0.48 (There could be different answers) Why do we calculate it?: - In real world, things tend to cluster. When generated, the clusters might not happen.

Task 5

Generate a graph for modelling real world situation.

Network of different types of road between given number of cities:

```ruby
class RoadNetworkGenerator
  ROAD_TYPES = [:highway, :regular, :dirt]

  def initialize(cities_count, road_quality, average_distance)
    @cities_count = cities_count
    @road_quality = road_quality
    @average_distance = average_distance
  end

  def generate
    @road_network = Graph.new
    add_cities
    add_roads
    @road_network
  end

  private

  def add_cities
    @cities_count.times { |n| @road_network << City.new(n, "City ##{n}") }
  end

  def add_roads
    first, second = pick_random_cities
    road_type = pick_probable_road
    @road_network.add_edge(Road.new(first, second, road_type, road_length))
  end

  def pick_random_cities
    [@road_network[rand(cities_count)], @road_network[rand(cities_count)]]
  end

  def pick_probable_road
    ROAD_TYPES[rand(ROAD_TYPES.count)*@road_quality]
  end

  def road_length
    gaussian_rand(@average_distance, @average_distance*0.5, rand)
  end

  def gaussian_rand(mean, stddev, rand)
    theta = 2 * Math::PI * rand.call
    rho = Math.sqrt(-2 * Math.log(1 - rand.call))
    scale = stddev * rho
    x = mean + scale * Math.cos(theta)
    y = mean + scale * Math.sin(theta)
    return x, y
  end
```

```
end
```

Task 6

- calculate IN, OUT (how many with 1 in, etc)