Week 14 - Social Network Graphs 2

James Quacinella

11/15/2015

Exercise 10.5.1 (section 10.5.5)

Suppose graphs are generated by picking a probability p and choosing each edge independently with probability p, as in Example 10.21. For the graph of Fig. 10.20, what value of p gives the maximum likelihood of seeing that graph? What is the probability this graph is generated?

Answer

There are 6 possible edges, with 4 of them in the graph, between 4 nodes. Therefore, we have probability p for 4 edges, and $1 - p$ for the other 2 missing edges. The probability of the given graph is written as:

$$P(G) = (p^4) * (1 - p)^2$$

To find the p that gives a maximum for this function, we use standard calculus:

$$P'(G) = 4p^3(1 - p)^2 - 2p^4(1 - p)$$
$$0 = 4p^3(1 - p)^2 - 2p^4(1 - p)$$

Lets factor out a $2p^3$:

$$0 = 2p^3(2(1 - p)^2 - p(1 - p))$$

Lets drop the outer term, since that is only 0 when p = 0.

$$0 = 2(1 - p)^2 - p(1 - p)$$
$$0 = 2(1 - 2p + p^2) - p + p^2)$$
$$0 = 2 - 4p + 2p^2 - p + p^2$$
$$0 = 3p^2 - 5p + 2$$
$$0 = (3p - 3)(p - \frac{2}{3})$$

This insinuates that p = 1 is another place where the function reaches a minimum. Our maximum is therefore $p = \frac{2}{3}$. The probability associated with p is:

$$P(G) = (\frac{2}{3})^4 * (1 - \frac{2}{3})^2$$
$$P(G) = (\frac{2^4}{3^4}) * (\frac{1}{3})^2$$
$$P(G) = (\frac{16}{81}) * (\frac{1}{9})$$
$$P(G) = (\frac{16}{81}) * (\frac{1}{9})$$
$$P(G) \approx 0.02194787$$

Use R for 10.7.1 (section 10.7.6)

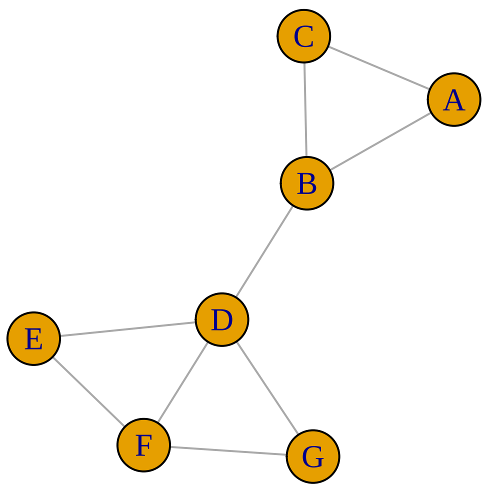
How many triangles are there in the graphs: (a) Figure 10.1. (b) Figure 10.9. (c) Figure 10.2.

Answer

```
find_triangles <- function(g, verbose=TRUE) {
  edges <- E(g);
  for (i in 1:length(edges)) {
    nodes <- get.edges(g, edges[i]);

    v1 <- nodes[1];
    v2 <- nodes[2];

    # This emulates a index lookup to get list of neighboring nodes#
    u <- neighborhood(g, 1, nodes=v1)[[1]];

    # For each potential triable betwene v1, v2 and ui
    for (ui in u) {
      # Check if ui and v2 do have a link, and make sure ordering is correct
      if(g[ui, v2] == 1 && v1 < ui && v1 < v2 && v2 < ui) {
        if(verbose) {
          print(paste("Triangle found", get.vertex.attribute(g, "name", v1),
                                        get.vertex.attribute(g, "name", v2),
                                        get.vertex.attribute(g, "name", ui)));
        }
      }
    }
  }
}
```

(a) Fig 10.1

```r
input_edges <- c(1,2, 2,3, 1,3, 2,4, 4,5, 4,6, 4,7, 6,7, 5,6);
g <- graph(input_edges, n=max(input_edges), directed=FALSE);
g <- set.vertex.attribute(g, "name", 1, "A");
g <- set.vertex.attribute(g, "name", 2, "B");
g <- set.vertex.attribute(g, "name", 3, "C");
g <- set.vertex.attribute(g, "name", 4, "D");
g <- set.vertex.attribute(g, "name", 5, "E");
g <- set.vertex.attribute(g, "name", 6, "F");
g <- set.vertex.attribute(g, "name", 7, "G");
g <- set.vertex.attribute(g, "name", 8, "H");
g <- set.vertex.attribute(g, "name", 9, "I");
plot(g, layout=layout.kamada.kawai, vertex.size=25);
```



Lets run the algo and see the performance:

```r
ptm <- proc.time() # Start the clock!
find_triangles(g);
```

```
## [1] "Triangle found A B C"
## [1] "Triangle found D E F"
## [1] "Triangle found D F G"
```

```r
print(proc.time() - ptm)
```

```
##    user  system elapsed
##   1.196   0.039   1.236
```

Not too sure why, but the initial run is always slow. This must be R lazy loading some libraries / code, so lets run it again:

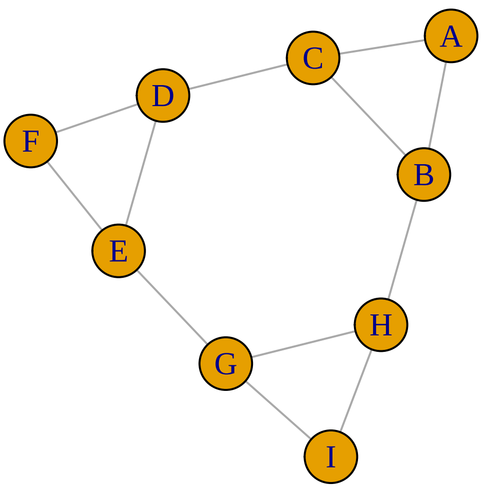
```r
num_iter <- 10;

for(i in seq(num_iter)) {
  ptm <- proc.time()
```

```r
  find_triangles(g, verbose=FALSE);
  print(proc.time() - ptm)
}
```

```
##    user  system elapsed
##   0.101   0.003   0.105
##    user  system elapsed
##   0.087   0.000   0.087
##    user  system elapsed
##   0.087   0.000   0.087
##    user  system elapsed
##   0.085   0.000   0.085
##    user  system elapsed
##    0.09    0.00    0.09
##    user  system elapsed
##   0.086   0.000   0.086
##    user  system elapsed
##   0.091   0.000   0.091
##    user  system elapsed
##   0.086   0.000   0.086
##    user  system elapsed
##   0.085   0.000   0.085
##    user  system elapsed
##   0.086   0.000   0.086
```

(b) Fig 10.9

```
input_edges <- c(1,2, 2,3, 1,3, 3,4, 4,5, 4,6, 5,6, 2,8, 8,9, 7,8, 5,7, 7,9);
g <- graph(input_edges, n=max(input_edges), directed=FALSE);
g <- set.vertex.attribute(g, "name", 1, "A");
g <- set.vertex.attribute(g, "name", 2, "B");
g <- set.vertex.attribute(g, "name", 3, "C");
g <- set.vertex.attribute(g, "name", 4, "D");
g <- set.vertex.attribute(g, "name", 5, "E");
g <- set.vertex.attribute(g, "name", 6, "F");
g <- set.vertex.attribute(g, "name", 7, "G");
g <- set.vertex.attribute(g, "name", 8, "H");
g <- set.vertex.attribute(g, "name", 9, "I");
plot(g, layout=layout.kamada.kawai, vertex.size=25);
```



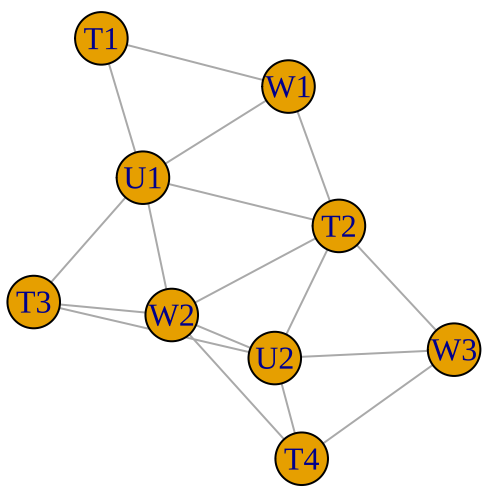

```
num_iter <- 10;

for(i in seq(num_iter)) {
  ptm <- proc.time()
  find_triangles(g, verbose=FALSE);
  print(proc.time() - ptm)
}
```

```
##    user  system elapsed
##   0.141   0.000   0.140
##    user  system elapsed
##   0.112   0.000   0.111
##    user  system elapsed
##   0.112   0.000   0.114
##    user  system elapsed
##   0.123   0.000   0.123
##    user  system elapsed
##   0.112   0.000   0.112
##    user  system elapsed
##   0.114   0.000   0.114
##    user  system elapsed
##   0.115   0.000   0.115
```

```
##    user  system elapsed 
##   0.111   0.000   0.111 
##    user  system elapsed 
##   0.112   0.000   0.112 
##    user  system elapsed 
##   0.113   0.000   0.112 
```

(c) Fig 10.2

```r
input_edges <- c(1,3, 1,4, 1,5, 1,7, 1,8, 2,4, 2,5, 2,6, 2,8, 2,9, 3,7, 4,7, 4,8, 4,9, 5,8, 6,8, 6,9)
g <- graph(input_edges, n=max(input_edges), directed=FALSE)
g <- set.vertex.attribute(g, "name", 1, "U1")
g <- set.vertex.attribute(g, "name", 2, "U2")
g <- set.vertex.attribute(g, "name", 3, "T1")
g <- set.vertex.attribute(g, "name", 4, "T2")
g <- set.vertex.attribute(g, "name", 5, "T3")
g <- set.vertex.attribute(g, "name", 6, "T4")
g <- set.vertex.attribute(g, "name", 7, "W1")
g <- set.vertex.attribute(g, "name", 8, "W2")
g <- set.vertex.attribute(g, "name", 9, "W3")
plot(g, layout=layout.kamada.kawai, vertex.size=25)
```




```r
num_iter <- 10;

for(i in seq(num_iter)) {
  ptm <- proc.time()
  find_triangles(g, verbose=FALSE);
  print(proc.time() - ptm)
}
```

```
##    user  system elapsed
##   0.238   0.000   0.238
##    user  system elapsed
##   0.225   0.000   0.225
##    user  system elapsed
##    0.22    0.00    0.22
##    user  system elapsed
##   0.225   0.000   0.225
##    user  system elapsed
##   0.223   0.000   0.224
##    user  system elapsed
##   0.220   0.000   0.221
##    user  system elapsed
##   0.225   0.000   0.225
```

```
##    user  system elapsed
##   0.221   0.000   0.221
##    user  system elapsed
##   0.225   0.000   0.225
##    user  system elapsed
##   0.222   0.000   0.222
```