INFX 576: Problem Set 7 - Network Dynamics[*]

Jay Chauhan

Due: Thursday, March 2, 2017

Collaborators: Avanti Chande, Gosuddin Siddiqi

Instructions:

Before beginning this assignment, please ensure you have access to R and RStudio.

1. Download the `problemset7.Rmd` file from Canvas.
2. Replace the "Insert Your Name Here" text in the `author:` field with your own full name. Any collaborators must be listed on the top of your assignment.
3. Be sure to include well-documented (e.g. commented) code chucks, figures and clearly written text chunk explanations as necessary. Any figures should be clearly labeled and appropriately referenced within the text.
4. Collaboration on problem sets is acceptable, and even encouraged, but each student must turn in an individual write-up in his or her own words and his or her own work. The names of all collaborators must be listed on each assignment. Do not copy-and-paste from other students' responses or code.
5. When you have completed the assignment and have **checked** that your code both runs in the Console and knits correctly when you click `Knit PDF`, rename the R Markdown file to `YourLastName_YourFirstName_ps7.Rmd`, knit a PDF and submit the PDF file on Canvas.

Setup:

In this problem set you will need, at minimum, the following R packages.

```r
# Load standard libraries
library(statnet)
```

Problem 1: Social Processes and Structure

Imagine a social group comprised of individuals from two cultures: the "reds" and the "blues." Both groups are alike in their desire for transitive friendships; however, they differ in how they react to intransitivity.

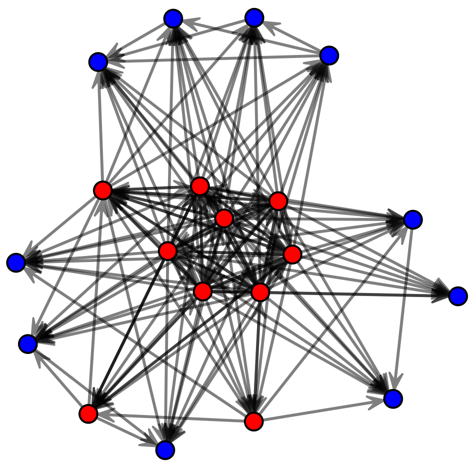
When placed in an arbitrary network, reds tend to act by extending friendship towards those with whom they have a transitive precondition. In other words, the condition $v_i \to v_j$ and $v_j \to v_k$, generates $v_i \to v_k$. Blues, in contrast, generally respond by severing ties which are associated with uncompleted transitive states. In other words, $v_i \to v_j \to v_k$, leads to $v_i \not\to v_j$.

To simulate the above process on a network with `nred` "red" and `nblue` "blue" vertices, use the following function:

```r
transsim<-function(nred,nblue,action.prob.red=c(0.9,0.05,0.05),
                   action.prob.blue=c(0.05,0.9,0.05),initial.density=0.2,
                   plot.result=TRUE,return.graph=FALSE,...){
  n<-nred+nblue
  #Draw the initial graph
  g<-rgraph(n,tp=initial.density)
```

[*]Problems originally written by C.T. Butts (2009)

```r
  rownames(g)<-c(rep("r",nred),rep("b",nblue))
  colnames(g)<-c(rep("r",nred),rep("b",nblue))
  #Now, update in a random order
  uo<-sample(1:n)
  for(i in uo){
    for(j in 1:n)
      for(k in 1:n)
        if((i!=j)&&(j!=k)&&(i!=k)){
          if((g[i,j]*g[j,k])&&!g[i,k]){
            if(i<=nred)
              action<-sample(1:3,1,prob=action.prob.red)
            else
              action<-sample(1:3,1,prob=action.prob.blue)
            if(action==1)
              g[i,k]<-1
            else if(action==2)
              g[i,j]<-0
          }
        }
  }
  #Plot the graph, if necessary
  if(plot.result)
    gplot(g,vertex.col=c(rep(2,nred),rep(4,nblue)),edge.col=rgb(0,0,0,.5))
  #Return, if needed
  if(return.graph)
    return(g)
  invisible()
}

transsim(10,10)
```



The above function should produce a plot of the resulting network. You can return the adjacency matrix by setting the `return.graph` argument to be `TRUE`. The first N vertices will be the red ones, followed by the blue ones.

You can change the probabilities of adding an edge, dropping an edge, or taking no action (respectively) in response to intransitivity for each group with the `action.prob.red` and `action.prob.blue` arguments. For example setting `action.prob.red=c(0.9, 0.05, 0.05)` will set reds to response to intransitivity by adding an edge 90% of the time, dropping an edge 5% of the time, and ignoring the violation 5% of the time. Notice this is the default setting.

`initial.density` is the initial density of the network. `nred` and `nblue` give the number of nodes in each group.

(a) Prediction

Consider what might happen when persons from both groups are placed together in a single friendship network. Make an initial guess at what you think will happen. • When there is a friendship which is initiated by a blue person to either a blue or a red person, and if the second person has a tie with a third person, again irrespective of the third person being a blue or a red, then the blue person's ties fall thorugh.

- For all other cases, the friendship remains intact, and there would be a transitive relation.

(b) Simulation

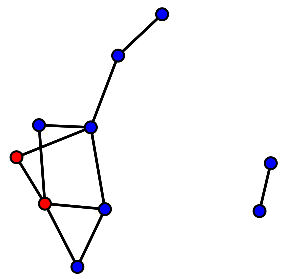
Design a simple simulation study using the `transsim` function. Consider a group of 10 red nodes and 10 blue nodes. After your experiment you should answer the following questions:

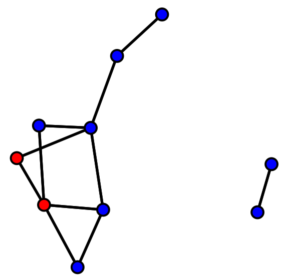
- What can you conclude about the structural effects of responses to intransitivity on the red and blue nodes? • Here we can say that when intrasitivity was introduced in the model because of the blues, it might have lead to the decrease in the overall density of the network. If the intransitivity was not introduced, there would have been a much denser network and in general a large order of interaction between the two groups.
- How is the total structure affected? • The overall structured is heavily concentrated on the red nodes and sparsely on the blue node, which is reflected in its density.
- How might you describe the overall state which tends to result from the combination of red and blue behaviors? • From the above network plots, we might be able to say that theere is a core-preiphery structure that is formed, with the red nodes being at the core having a dense network amongst themselves, while the blue nodes acting as the periphery nodes.
- How are these conclusions similar to or different from your initial guess about the outcome of the social process? • Our initial assumption was whenever blue initiates a friendship, then most of the times the ties would break because of the intrasitive nature of the blue nodes, while in cases where the reds initiates the friendship and extends it via transitive ties, there is a corresponding increase in these densities in the graph. Thus the initial assumpstion was pretty much confirmed by the network

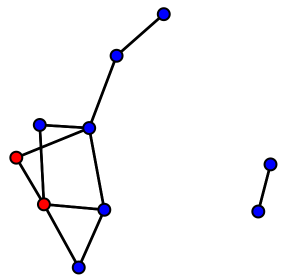
Problem 2: Network Diffusion

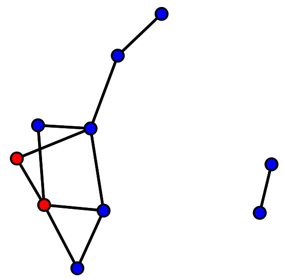
The following is a simple function to simulate cascading behavior in a social network. Write detailed comments for this code to demonstrate you understand the function. Feel free to modify or adjust the functionality.

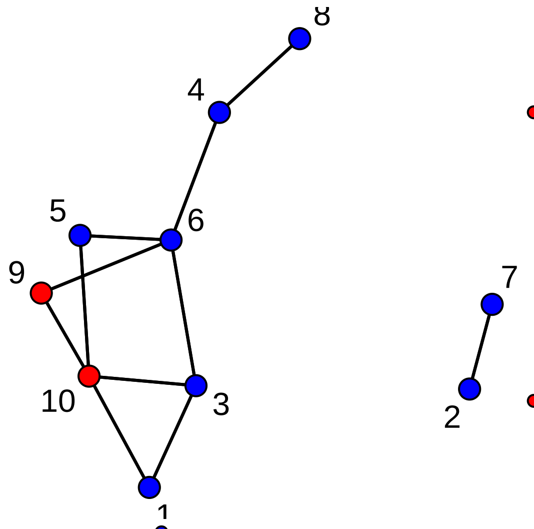
```r
#Simulating cascading behavior with default values

diffusion <- function(network_size=10, network_density=0.2, b_payout=3,
  a_payout=2, num_seeds=2, max_steps=10){

  # create a random graph with the network given sze and density
  g <- network(rgraph(network_size, tprob=network_density), directed=FALSE)

  #assign color to all the vertex in the nodes
  vertex_colors <- rep("blue", network_size)

  #take a random sample of nodes as the intial nodes of A and color them red
  initial_nodes <- sample(1:network_size, num_seeds)
  vertex_colors[initial_nodes] <- "red"
  # take the coordinates of the plot to reproduce it
  coords <- gplot(g, usearrows=FALSE, vertex.col=vertex_colors, displaylabel=TRUE)

  #calculate the threshold value to be set for conversion
  q = b_payout / (b_payout + a_payout)

  par(mar=c(0,0,0,0), mfrow=c(3,2))

  step = 1

  #loop the cascade till the number of steps is equal to a predetermined number of steps or till all th
  while(step < max_steps && sum(vertex_colors=="red") < network_size){
  for (i in 1:network_size){

  #retrieve all the neigbors of the current node
  neighborInds <- get.neighborhood(g, i, type="combined")

  #retrieve all the sum of all red neighbors of the current node
  obsA <- sum(vertex_colors[neighborInds]=="red")

  #compare the fraction of the red nodes to the threshold and convert the current node to red if the fr
  if (obsA/length(neighborInds) >= q){
  vertex_colors[i] <- "red"
  }
  gplot(g, usearrows=FALSE, vertex.col=vertex_colors, coord=coords)
  Sys.sleep(.2)
  step <- step + 1
  }
  }
}
diffusion()
```

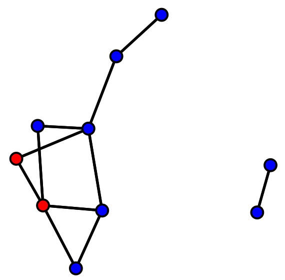


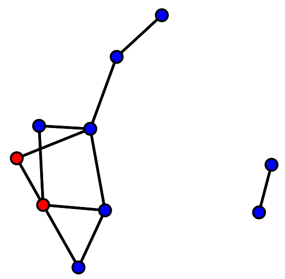


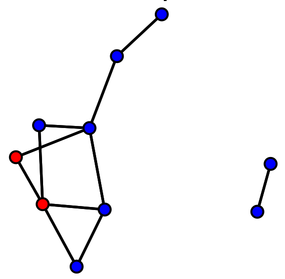


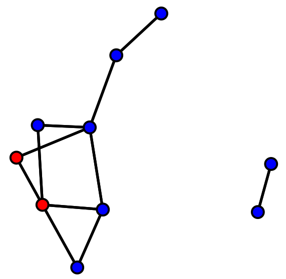


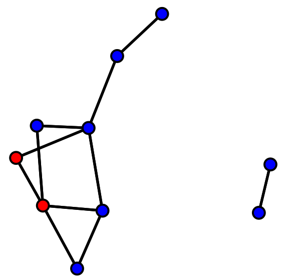


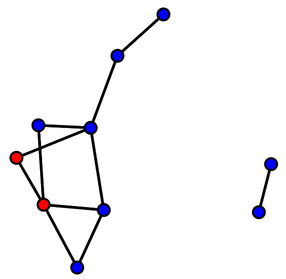




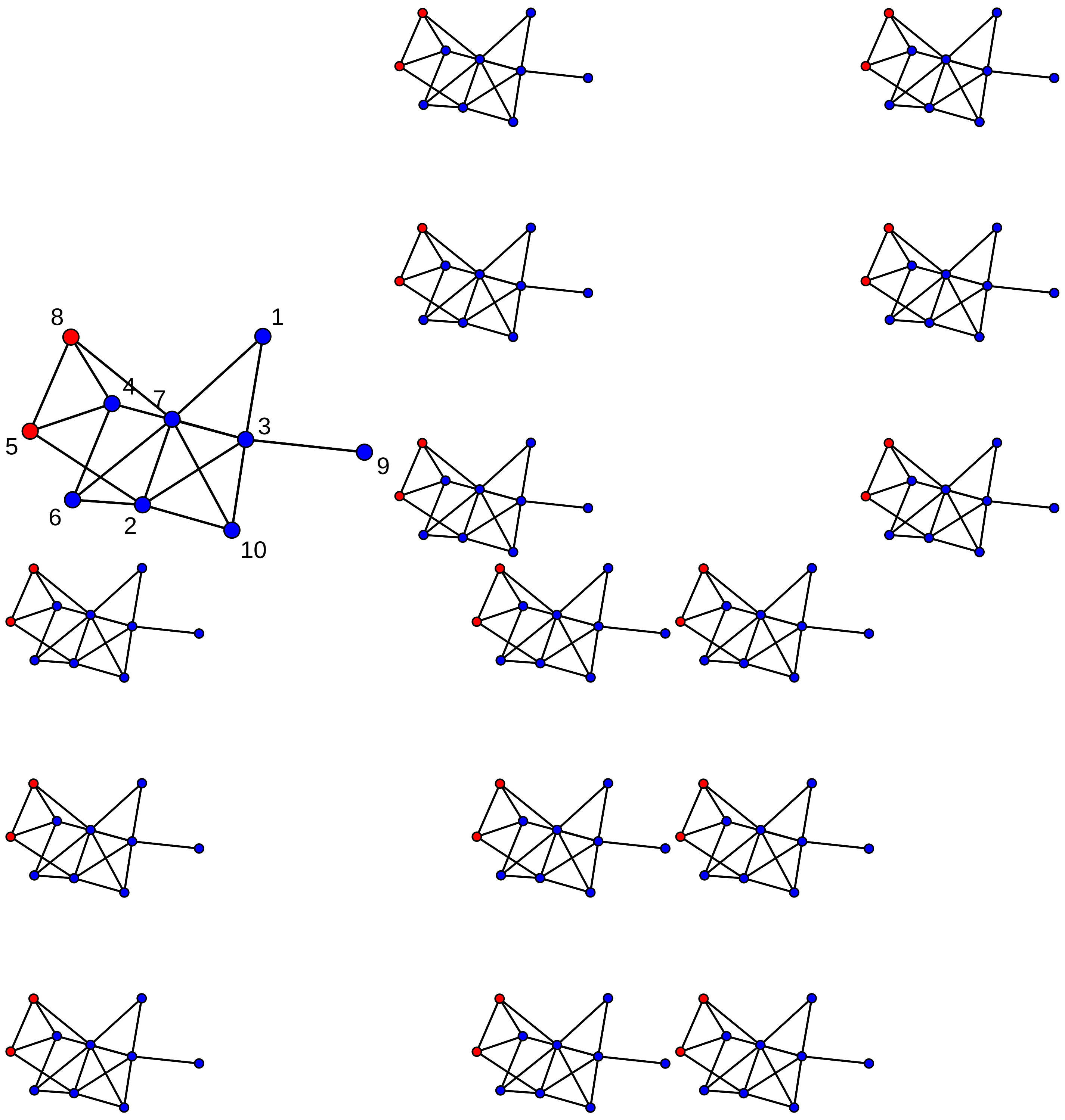


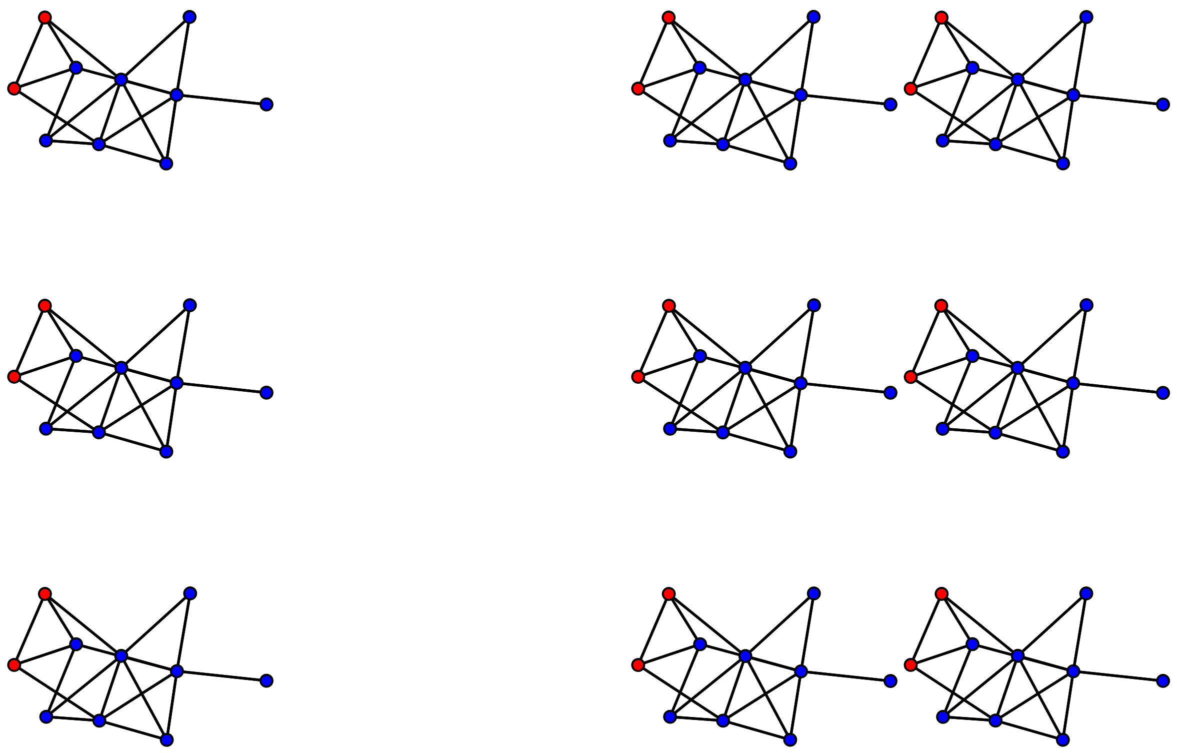




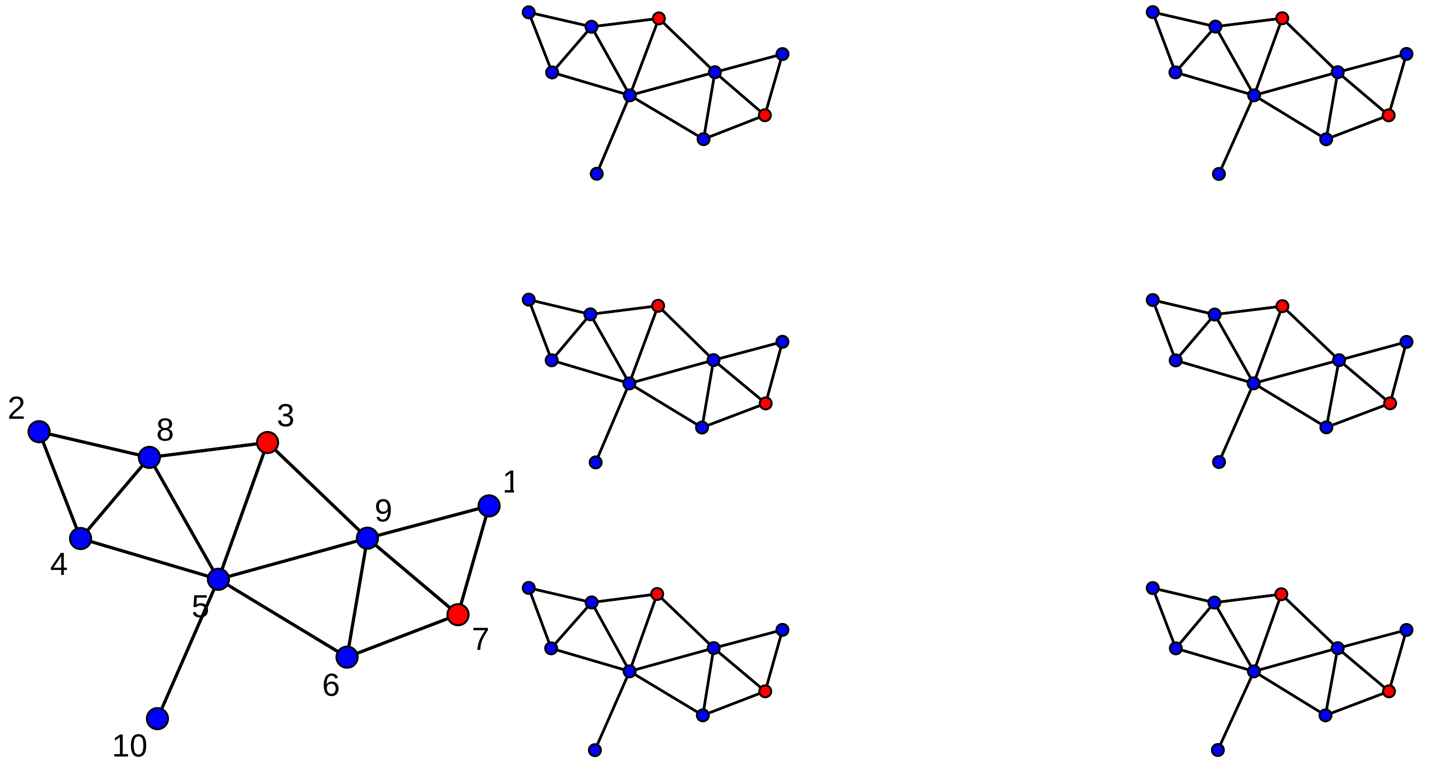


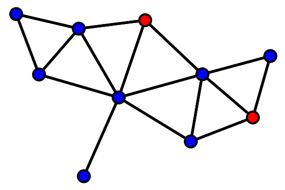
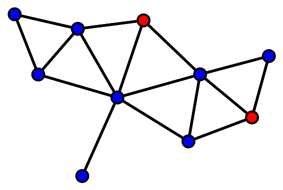
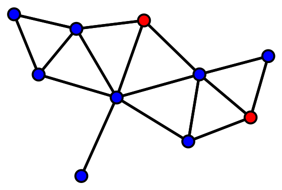
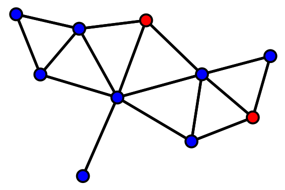
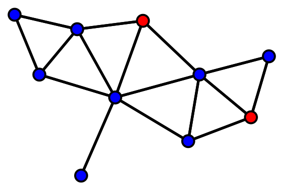
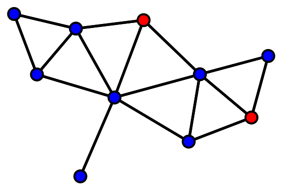
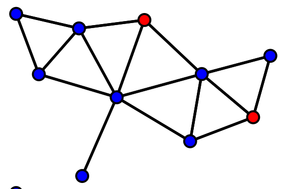
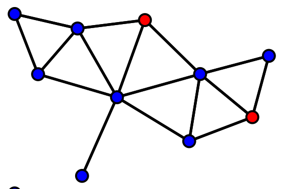
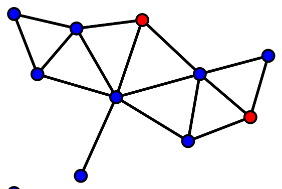
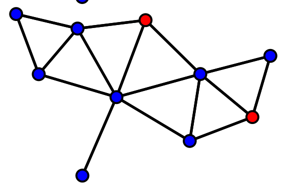
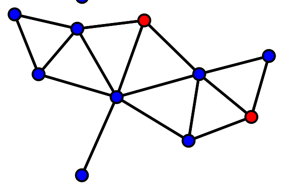
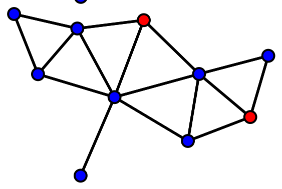
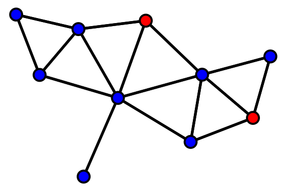
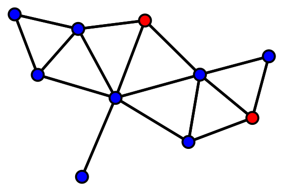
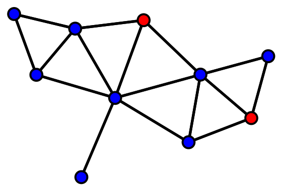
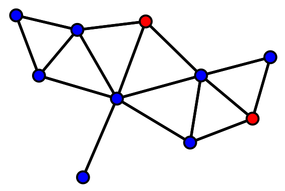
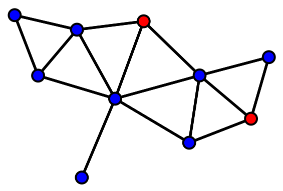
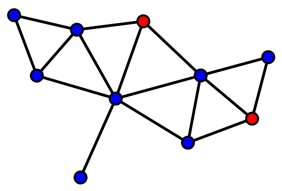
```
#Simulating cascading behavior with increased max steps values

diffusion2 <- function(network_size=10, network_density=0.2, b_payout=3,
  a_payout=2, num_seeds=2, max_steps=25){

  # create a random graph with the network given sze and density
  g <- network(rgraph(network_size, tprob=network_density), directed=FALSE)

  #assign color to all the vertex in the nodes
  vertex_colors <- rep("blue", network_size)

  #take a random sample of nodes as the intial nodes of A and color them red
  initial_nodes <- sample(1:network_size, num_seeds)
  vertex_colors[initial_nodes] <- "red"

  # take the coordinates of the plot to reproduce it
  coords <- gplot(g, usearrows=FALSE, vertex.col=vertex_colors, displaylabel=TRUE)

  #calculate the threshold value to be set for conversion
  q = b_payout / (b_payout + a_payout)

  par(mar=c(0,0,0,0), mfrow=c(3,2))

  step = 1

  #loop the cascade till the number of steps is equal to a predetermined number of steps or till all th
  while(step < max_steps && sum(vertex_colors=="red") < network_size){
  for (i in 1:network_size){

  #retrieve all the neigbors of the current node
  neighborInds <- get.neighborhood(g, i, type="combined")

  #retrieve all the sum of all red neighbors of the current node
  obsA <- sum(vertex_colors[neighborInds]=="red")

  #compare the fraction of the red nodes to the threshold and convert the current node to red if the fr
  if (obsA/length(neighborInds) >= q){
  vertex_colors[i] <- "red"
  }
  gplot(g, usearrows=FALSE, vertex.col=vertex_colors, coord=coords)
  Sys.sleep(.2)
  step <- step + 1
  }
  }
}
diffusion2()
```


8
1
4
7
3
5
9
6
2
10



```r
#Simulating cascading behavior when changing the threshold

diffusion3 <- function(network_size=10, network_density=0.2, b_payout=9,
  a_payout=1, num_seeds=2, max_steps=25){

  # create a random graph with the network given sze and density
  g <- network(rgraph(network_size, tprob=network_density), directed=FALSE)

  #assign color to all the vertex in the nodes
  vertex_colors <- rep("blue", network_size)

  #take a random sample of nodes as the intial nodes of A and color them red
  initial_nodes <- sample(1:network_size, num_seeds)
  vertex_colors[initial_nodes] <- "red"

  # take the coordinates of the plot to reproduce it
  coords <- gplot(g, usearrows=FALSE, vertex.col=vertex_colors, displaylabel=TRUE)

  #calculate the threshold value to be set for conversion
  q = b_payout / (b_payout + a_payout)

  par(mar=c(0,0,0,0), mfrow=c(3,2))

  step = 1

  #loop the cascade till the number of steps is equal to a predetermined number of steps or till all th
  while(step < max_steps && sum(vertex_colors=="red") < network_size){
  for (i in 1:network_size){

  #retrieve all the neigbors of the current node
  neighborInds <- get.neighborhood(g, i, type="combined")

  #retrieve all the sum of all red neighbors of the current node
  obsA <- sum(vertex_colors[neighborInds]=="red")

  #compare the fraction of the red nodes to the threshold and convert the current node to red if the fr
  if (obsA/length(neighborInds) >= q){
  vertex_colors[i] <- "red"
  }
  gplot(g, usearrows=FALSE, vertex.col=vertex_colors, coord=coords)
  Sys.sleep(.2)
  step <- step + 1
  }
  }
}
diffusion3()
```



Experiment with this function to answer the following questions:

- When does this simulation result in everyone switching to A? • The simulation results in everyone switching to A when the early adopters are connected to a larger fraction of nodes and also when threshold value that is set for conversion is smaller. The simulation might also result in everyone switching to A when we increase the max steps for the simulation which results in a node being visited for conversion more than once. There is also a good chance that everyone would switch to A when the number of intial adopters increase.

- What causes the spread of A to stop? The spread of A stops when the early adopters of A are connected to a smaller proportion of nodes and also when the threshold value that is set for the conversoion is larger. The simulation might also cause the stop of A when the maximum step size is less, that is there is not enough steps to allow the entire network to convert