How Many People will read Trump's Next Post?

Group Member
3035467752 Lai Pak Hei Syrus 3035123655 Choi jae won
3035164283 Park Chan Ho 3035381104 Lam Tsun Ho Kevin

1. Introduction

Nowadays, social networking has become a part of our daily life. Sites such as Facebook or Twitter allow us to spread ideas more conveniently and effectively. After you write a post, your friends will see it when they log in. Some of them may even decide to share your post. This way, your idea will spread further. We found this an interesting topic, so we would like to study the effects of different factors on the spread of information.

The theory of complexity can be applied to this phenomenon. There are several similarities between the spread of information on social networking sites and the germ model. There is no central command in either system, and the process is determined by a few basic rules and initial conditions.

The US president Donald Trump's posts have been chosen as the subject in our simulation model, in order to apply it to real-life situations. We would like to study its spread among the population in the framework of the complexity theory.

2. Background

- **Complex contagions**

Information flows, such as the posts from Facebook, Instagram, Twitter or any other social networking sites, can be expressed by the term "complex contagions". Complex contagions discussed how the spread of information or memes will be, like the spread of virus. Nonetheless, unlike the spread of virus, the virality of these memes are not affected by the distances between individuals, but social reinforcement and homophily instead.[1]

- **Social reinforcement**

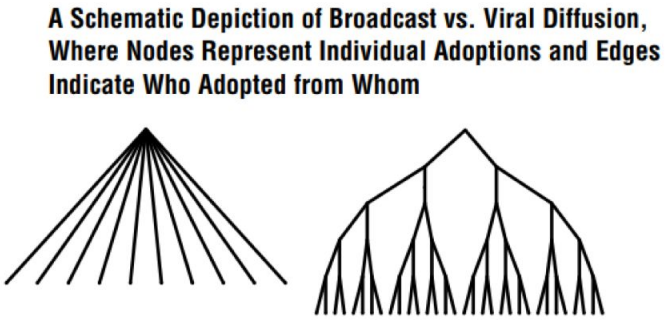
When individuals come together, they form small communities. If one gets exposed to some memes, it is likely that they will share the information with other members in the same community, and hence resulting in multiple exposures[1], and more members in the system gets exposed to that meme.

- **Homophily**

This is what varies between complex contagions and the spread of virus the most. Both of them are related to the closeness between the individuals, but it is more about the closeness in terms of the interpersonal relationship instead of the geographical one. Just like we will do so with our family members and friends than with strangers or those who we do not familiar with, people in the same community used to share similar ideas or the way of thinking, so it is more likely for them to share the memes within the community than with the members in other communities.[2,3]

- **Broadcast vs. Viral posts**

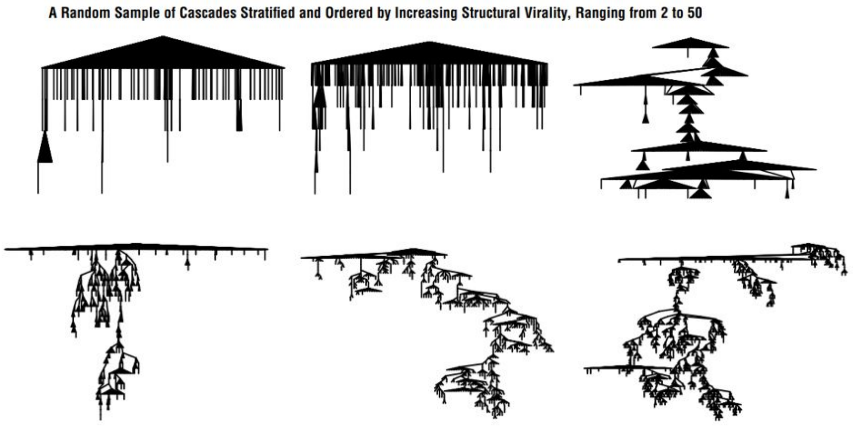
There are two types of memes, one is broadcasts, while another one is viral posts. In Figure 1, each line represents the connections between every two individuals, while each intersection represents a node in the system, all nodes lying in the same horizontal level is known as a 'generation'. For broadcasts, like when we are watching TV or listening to the radio broadcasts, most of the individuals should have received the same piece of information from the same source, the advertisement on TV or the radio broadcast. While for viral posts, most of the individuals will have received the meme from the previous generation but not from the same source. Since the viral post involves more interpersonal interactions while the broadcast mainly involves the interactions between the first generation and the root (i.e. the source), the structure of the graph for viral post gives a more complex and branching structure (i.e. structural virality) than a regular structure as that of the broadcast does.



<Figure 1: Diagrams for the diffusion of broadcast and viral post >[2]

- **Diffusion trees**

Derived from and extending Figure 1, the diffusion trees is introduced. It shows how the information flows among the individuals in the system. From the left to the right, and from the top to the bottom will we see the ordering of the graphs according to the virality of the meme in Figure 2, which means from broadcast to viral post.[2] Diffusion trees are run by only two simple rules, 'to share' or 'not to share' the information. If they do not, the information flow (i.e. diffusion) stops; If they do, the diffusion continues across the generations. Similar to the previous graph, a large proportion of individuals receive the information from the same root, as it is expected that the meme has already been widely spread, so that most of the individuals will not keep on sharing the meme to others. Therefore, the diffusion trees for broadcast stops in the first few generations. At the meantime, each generation should consider whether to share the meme or not, and the upcoming generations will have to do so, just as the previous generations did. Hence, by reiterations, we expect the latter generations should give a more complex structure, for the question of whether to share the piece of information or not is absolutely a personal choice without central control and somehow a random and unpredictable process.



<Figure 2: Diffusion Tree> [2]

In the light of this, information flow which involves interpersonal interactions is a complex system.

To have known that information flow is a complex system, we wanted to visualise a particular information flowing within a large number of people. With such a motivation we have created a simulation. In terms of the report, we have viewed each one and one's relationship between other people to be a viral post structure. So we intended to visualise at how information diffuses within a viral post structure over time.

In order to do so, each person is represented as nodes; and the relationship between one person and his friends is represented as links between nodes. One person posting a particular information on their own Facebook or twitter account will allow his friends to see this post. This process will iterate if one's friend decides to repost this information. And as time passes reiteration of this process is how information spreads across people.

To talk about the general assumption that we have taken in order to create a simulation of our own, there were four assumptions.

A. We have only taken into account of people reposting to be the only mode of transferring information. Because, in reality, there are agents like the media or other form of broadcasting allows information to flow more vastly; hence we have adjusted the initial extent of information outbreak to make our simulation more practical.
B. The number of friends each person has to be random but have an mean. This to understand how sensitive average number of friends in respect to the output. Furthermore this randomness allows the simulation to be more close to reality.
C. Everyone will see the post by their friend. Though in real life we may not see all the posts by our friends due to various reasons like not accessing sns often or some other reason, in order to create a simulation we have reduced such cases.
D. We have assumed there is only one post spreading across the system.

Based on these assumptions, we have chosen four initial inputs.

A. Initial number of people who repost Trump's post (i.e. Initial outbreak of information)
B. Average number of friends (i.e. Average node degree)
C. Probability of one sharing the post
D. Total number of nodes in the system

The reason behind choosing these four inputs were because we thought these were the most important four factors in terms of information flow. With these inputs, our expected output is as the following:

A. Number/Proportion of people in the network who has came across the information
B. Which initial input is most sensitive in respect to the output

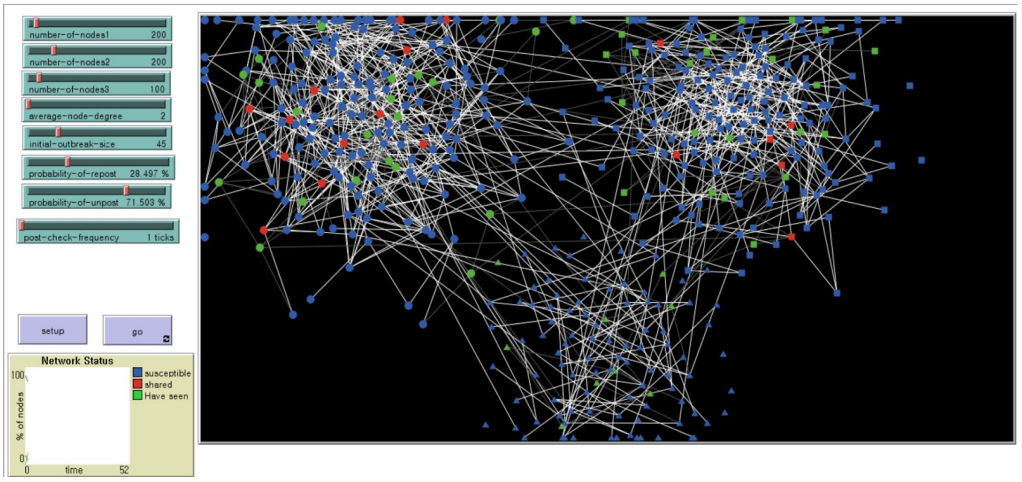
In order to examine the most sensitive input, we have set of default initial inputs. The default inputs are

- Number of Node = 500
- Average Node Degree = 1%
- Initial outbreak size = 5%
- Probability of repost = 30%
- Probability of not post = 70% (1 - probability of repost)

Figures of probability of repost (in reality both retweeting and replying) are derived from our research on Twitter feeds.[4] Moreover initial outbreak size and average node degree are amplified numbers because in reality there are more modes of communication; in fact if we were to simulate with figures derived just from Twitter, there will be extensively limited spread of information.

3. Simulation

For our simulation, we modified the codes in NetLogo which is called virus on network.



<Figure 3: Network simulation model>

There are three types of nodes in three different clusters. Each of Circle, Square, and Triangle nodes represents three different clusters. Red node(Shared) represents a person who has shared the post. Green node(Have seen) represents a person who already decided not to share or who already shared the post. Blue node(Susceptible) represents a person who has not seen the post yet.

- **Flow of model**

(1) At initial stage, there are only red and blue nodes representing people who shared the post for the first time and the people who have not seen the post yet respectively.
(2) Under some probability, neighbor nodes of red nodes choose to share the post or not. If the node chooses to share the post, it will turn red, and if not, it will turn green.
(3) Nodes that were previously red would then turn green because they already have seen the post and decided whether to share the post or not (People who have shared the post are assumed to have seen the post).
(4) We iterate this process till there are no more red nodes in the program, which means there is no more spread of information.
(5) After program ends, there will be only two types of nodes; green and blue, each representing people who have seen the post and who have not seen the post respectively.

- **Default constants and logic of code related with reality**

There are the default constants of our simulation, which we take as initial conditions of our model. Each of number of nodes 1,2 and 3 represents the total number of people in each clusters. Average node degree is average number of friends of a node. Initial outbreak size is number of people who have shared the post at the very first stage. Probability of repost is the probability that a person shares the post and probability of unpost is the probability that a person would not share the post which must be equal to one minus probability of repost.

```
to go
  set counter 1
  if all? turtles [not shared?]
    [ stop ]
  ask turtles
  [
      set post-check-timer post-check-timer + 1
      if post-check-timer >= post-check-frequency
        [ set counter counter + 1 ]
  ]
  spread-virus
  spread-virus2
  do-virus-checks
  set probability-of-unpost probability-of-unpost + rate * counter
  set probability-of-repost probability-of-repost - rate * counter
  tick
end
```

<Figure 4: Part of codes of the simulation>

To make our model to be more realistic, we made probability of repost decreases and probability of unpost increases accordingly as time elapses because people are less likely to share the post long after the post is initially shard.(Figure 4).

```
to setup-connection-between-networks

  while [count links < (average-node-degree * number-of-nodes1 + constant)]
  [
    ask one-of turtles
    [
      let choice(one-of (other turtles with [networktype != 1]))
        if choice != nobody [ create-link-with choice]
    ]
  ]
end
```

<Figure 5: Part of codes of the simulation 2>

Also, we have set the probability of repost with people in same cluster to be higher than that with people in different cluster. (Figure 5).

- **Outcome**

By running simulation, we wanted to see how the total number of green nodes, which represent the total number of people who have seen the post, differ with different initial conditions.

Average node degree	Trial 1	Trial 2	Trial 3
3%	**12.8%**	**12.4%**	**11.2%**
5%	**17%**	**18.4%**	**16.2%**
7%	**22.6%**	**21.4%**	**18.4%**

Initial outbreak size	Trial 1	Trial 2	Trial 3
5%	**18.6%**	**19.4%**	**20.6%**
7%	**20%**	**24.8%**	**21.4%**
9%	**29.6%**	**27.2%**	**27%**

Probability of repost	Trial 1	Trial 2	Trial 3
15%	**8.6%**	**8.2%**	**10.2%**
25%	**25.8%**	**22.6%**	**25.2%**
35%	**25.8%**	**22.6%**	**25.2%**

4. Conclusion

- **Results**

From the results of the simulations as described above, we found that several factors have a significant effect on the total number of people who will see a Twitter post. The average node degree,

the initial outbreak size, and the probability of sharing are all positively proportional to the result. In contrast, the size of the population does not have any significant effect, with only some random fluctuations in the outcomes.

- **Discussion**

Our original question is: how many people are going to see Trump's next post. In our simulation model, the proportion would be around 10-20%. However, the values of the initial conditions in reality will be different from the ones chosen in the model (as discussed soon below in "Limitations"). Also, Twitter does not release the figure of the exact number of people who have seen a Twitter post but only the total number of reposting. These mean that at this stage, we cannot confirm whether our model can indeed reflect the reality. Nevertheless, while we are not certain of their absolute effects on the proportion of the population that see a post, this study has investigated the relative effects of different factors on it. We hope to shed some light on how different initial conditions will influence the the proportion of people that will see Donald Trump's next post.

- **Limitations**

There are several limitations to our study design. One is that the population is assumed to be a mere 500 in our simulation model. In reality, however, the total number of Twitter users have reached 330 million as of the third quarter of 2017. Similarly, the initial size and the probability of reposting in the model may deviate significantly from the real-world values, as they are chosen based on our subjective judgements. Another limitation is that, in our model, every node has the same number of node degree. On the contrary, the real world resembles a small-world network, where a few people are hugely influential while the majority have little influence. Also, as we all know, people are less likely to share a post as time elapses. The value we have chosen for this decrease in the probability of sharing may not correspond to the real-world value.

- **Suggestions for future research**

Due to the limitations in this project, further studies are needed to collect sufficient data on the real-life values of different factors and to confirm their precise effects on the proportion of the population that will see a certain Twitter post. Also, there should be some modifications to the simulation model in order to better reflect the reality, where the information network resembles a small-world one. Another possible topic is to apply the results for Twitter to other social networking sites, and eventually to flow of information on the Internet in general.

References:

1. The Structural Virality of Online Diffusion (Sharad Goel, Ashton Anderson, Jake Hofman, Duncan J. Watts, 2016)
2. McPherson, M., Lovin, L. & Cook, J. Birds of a feather: Homophily in social □networks. Annual Review of Sociology 27, 415–444 (2001)
3. Centola, D. An experimental study of homophily in the adoption of health □behavior. Science 334, 1269–1272 (2011)
4. Twitter-retweet-stats. (2017, February 09). Retrieved December 05, 2017, from https://sysomos.com/inside-twitter/twitter-retweet-stats/
5. Virality Prediction and Community Structure in Social Networks (Lilian Weng, Filippo Menczer & Yong-Yeol Ahn, 2013)
6. Statistica. 2017. Number of monthly active Twitter users worldwide from 1st quarter 2010 to 3rd quarter 2017 (in millions). Retrieved from https://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users

Appendix

Original code

```
turtles-own
[
  infected?           ;; if true, the turtle is infectious
  resistant?          ;; if true, the turtle can't be infected
  virus-check-timer   ;; number of ticks since this turtle's last virus-check
]

to setup
  clear-all
  setup-nodes
  setup-spatially-clustered-network
  ask n-of initial-outbreak-size turtles
    [ become-infected ]
  ask links [ set color white ]
  reset-ticks
end

to setup-nodes
  set-default-shape turtles "circle"
  create-turtles number-of-nodes
  [
    ; for visual reasons, we don't put any nodes *too* close to the edges
    setxy (random-xcor * 0.95) (random-ycor * 0.95)
    become-susceptible
    set virus-check-timer random virus-check-frequency
  ]
end
```

```
to setup-spatially-clustered-network
  let num-links (average-node-degree * number-of-nodes) / 2
  while [count links < num-links ]
  [
    ask one-of turtles
    [
      let choice (min-one-of (other turtles with [not link-neighbor? myself])
                   [distance myself])
      if choice != nobody [ create-link-with choice ]
    ]
  ]
  ; make the network look a little prettier
  repeat 10
  [
    layout-spring turtles links 0.3 (world-width / (sqrt number-of-nodes)) 1
  ]
end

to go
  if all? turtles [not infected?]
    [ stop ]
  ask turtles
  [
     set virus-check-timer virus-check-timer + 1
     if virus-check-timer >= virus-check-frequency
       [ set virus-check-timer 0 ]
  ]
  spread-virus
  do-virus-checks
  tick
end
```

```
60
61 to become-infected  ;; turtle procedure
62   set infected? true
63   set resistant? false
64   set color red
65 end
66
67 to become-susceptible  ;; turtle procedure
68   set infected? false
69   set resistant? false
70   set color blue
71 end
72
73 to become-resistant  ;; turtle procedure
74   set infected? false
75   set resistant? true
76   set color gray
77   ask my-links [ set color gray - 2 ]
78 end
79
```

```
79
80 to spread-virus
81   ask turtles with [infected?]
82     [ ask link-neighbors with [not resistant?]
83         [ if random-float 100 < virus-spread-chance
84             [ become-infected ] ] ]
85 end
86
87 to do-virus-checks
88   ask turtles with [infected? and virus-check-timer = 0]
89   [
90     if random 100 < recovery-chance
91     [
92       ifelse random 100 < gain-resistance-chance
93         [ become-resistant ]
94         [ become-susceptible ]
95     ]
96   ]
97 end
98
99
100 ; Copyright 2008 Uri Wilensky.
101 ; See Info tab for full copyright and license.
```

Modified Code

```
globals[
  counter
  tmp
]
turtles-own
[
  shared?             ;; if true, the turtle is infectious
  notshared?            ;; if true, the turtle can't be shared
  post-check-timer    ;; number of ticks since this turtle's last virus-check
  networktype
]

to setup
  clear-all
  ; setup first group
  setup-nodes1
  setup-spatially-clustered-network1

  ;setup second group
  setup-nodes2
  setup-spatially-clustered-network2

  ;setup third group
  setup-nodes3
  setup-spatially-clustered-network3

  setup-connection-between-networks1
  setup-connection-between-networks2
  setup-connection-between-networks3

  ask n-of initial-outbreak-size turtles
    [ become-shared ]
  ask links [ set color white ]
  reset-ticks
end
```

```
to setup-connection-between-networks1
  let num-interlinks (average-node-degree * number-of-nodes3) / 2 + (average-node-degree * number-of-nodes2) + (average-node-degree * number-of-nodes1) + 20
  while [count links < num-interlinks ]
  [
    ask one-of turtles with [networktype = 1]
    [
      let choice(one-of (other turtles with [networktype = 2]))
        if choice != nobody [ create-link-with choice]
    ]
  ]
end
to setup-connection-between-networks2
  let num-interlinks (average-node-degree * number-of-nodes3) / 2 + (average-node-degree * number-of-nodes2) + (average-node-degree * number-of-nodes1) + 40
  while [count links < num-interlinks]
  [
    ask one-of turtles with [networktype = 2]
    [
      let choice(one-of (other turtles with [networktype = 3]))
        if choice != nobody [ create-link-with choice]
    ]
  ]
end
to setup-connection-between-networks3
  let num-interlinks (average-node-degree * number-of-nodes3) / 2 + (average-node-degree * number-of-nodes2) + (average-node-degree * number-of-nodes1) + 60
  while [count links < num-interlinks]
  [
    ask one-of turtles with [networktype = 3]
    [
      let choice(one-of (other turtles with [networktype = 1]))
        if choice != nobody [ create-link-with choice]
    ]
  ]
end
```

```
; for first topology
to setup-nodes1
  set-default-shape turtles "circle"
  create-turtles number-of-nodes1
  [
    ; for visual reasons, we don't put any nodes *too* close to the edges
    setxy (-30) (15)
    become-susceptible
    set networktype 1
    set post-check-timer random post-check-frequency
  ]
end

to setup-spatially-clustered-network1
  let num-links (average-node-degree * number-of-nodes1) / 2
  while [count links < num-links ]
  [
    ask one-of turtles
    [
      let choice (min-one-of (other turtles with [not link-neighbor? myself])
                   [distance myself])
      if choice != nobody [ create-link-with choice ]
    ]
  ]

  ; make the network look a little prettier
  repeat 10
  [
    layout-spring turtles links 0.3 (world-width / (sqrt number-of-nodes1)) 1
  ]
end
```

```
to setup-nodes2
  set-default-shape turtles "square"
  create-turtles number-of-nodes2
  [
    ; for visual reasons, we don't put any nodes *too* close to the edges
    setxy (15) (15)
    become-susceptible
    set networktype 2
    set post-check-timer random post-check-frequency
  ]
end


to setup-spatially-clustered-network2
  let num-links (average-node-degree * number-of-nodes2) / 2 + (average-node-degree * number-of-nodes1)
  while [count links < num-links ]
  [
    ask one-of turtles
    [
      let choice (min-one-of (other turtles with [not link-neighbor? myself])
                   [distance myself])
      if choice != nobody [ create-link-with choice ]
    ]
  ]
  ; make the network look a little prettier
  repeat 10
  [
    layout-spring turtles links 0.3 (world-width / (sqrt number-of-nodes2)) 1
  ]
end
```

```
to setup-nodes3
  set-default-shape turtles "triangle"
  create-turtles number-of-nodes3
  [
    ; for visual reasons, we don't put any nodes *too* close to the edges
    setxy (0) (-15)
    become-susceptible
    set networktype 3
    set post-check-timer random post-check-frequency
  ]
end

to setup-spatially-clustered-network3
  let num-links (average-node-degree * number-of-nodes3) / 2 + (average-node-degree * number-of-nodes2) + (average-node-degree * number-of-nodes1)
  while [count links < num-links ]
  [
    ask one-of turtles
    [
      let choice (min-one-of (other turtles with [not link-neighbor? myself])
                   [distance myself])
      if choice != nobody [ create-link-with choice ]
    ]
  ]
  ; make the network look a little prettier
  repeat 10
  [
    layout-spring turtles links 0.3 (world-width / (sqrt number-of-nodes3)) 1
  ]
end

to setup-connection-between-networks

  while [count links < (average-node-degree * number-of-nodes1 + 20)]
  [
    ask one-of turtles
    [
      let choice(one-of (other turtles with [networktype != 1]))
        if choice != nobody [ create-link-with choice]
    ]
  ]
end
```

```
to go
  set counter 1
  if all? turtles [not shared?]
    [ stop ]
  ask turtles
  [
     set post-check-timer post-check-timer + 1
     if post-check-timer >= post-check-frequency
       [ set counter counter + 1 ]
  ]
  spread-virus
  spread-virus2
  do-virus-checks
  set probability-of-unpost probability-of-unpost + 0.03 * counter
  set probability-of-repost probability-of-repost - 0.03 * counter
  tick
end

to become-shared  ;; turtle procedure
  set shared? true
  set notshared? false
  set color red
end

to become-susceptible  ;; turtle procedure
  set shared? false
  set notshared? false
  set color blue
end

to become-notshared  ;; turtle procedure
  set shared? false
  set notshared? true
  set color green
  ask my-links [ set color gray - 2 ]
end
```

```
to spread-virus

  ask turtles with [shared?] ;spread between different cluster
    [ set tmp networktype
      ask link-neighbors with [not notshared? and  networktype != tmp]
      [    if random-float 100 < probability-of-repost - 10
            [ become-shared ]
      ]
  ]
end
to spread-virus2 ;spread between same cluster

  ask turtles with [shared?]
    [ set tmp networktype
      ask link-neighbors with [not notshared? and  networktype = tmp]
      [    if random-float 100 < probability-of-repost - 10
            [ become-shared ]
      ]
  ]
end
to do-virus-checks
  ask turtles with [shared?]
  [
      if random 100 < probability-of-unpost
        [ become-notshared ]

  ]
end
```