```
globals [
  infinity
  highlight-string
  ticktimer
  ticktimerswitch

  average-path-length-of-lattice
  average-path-length

  clustering-coefficient-of-lattice
  clustering-coefficient

  pollution
  pvar
  totgeneralhealth
  totvictimhealth
  totleaderhealth
  avggeneralhealth
  avgvictimhealth
  avgleaderhealth
  totgeneralcc
  totvictimcc
  totleadercc
  totgenerallinkneighbors
  totvictimlinkneighbors
  totleaderlinkneighbors
  avggeneralcc
  avgvictimcc
  avgleadercc
  generalcc
  leadercc
  victimcc
  avggenerallinkneighbors
  avgvictimlinkneighbors
  avgleaderlinkneighbors
  avggenerallinkneighbordelta
  avgvictimlinkneighbordelta
  avgleaderlinkneighbordelta
  deadcircles
  deadstars
  deadtriangles
  tothouseholds
  totcircles
  tottriangles
  totstars
  totsquares
  generaldr
  victimdr
  faccuserdr
  pctvictims
  timetoritual
  timetoritualswitch
  ritualtime
  timetoleader
  timetoleaderswitch
]

breed [households household]
breed [wells well]
breed [deadhouseholds deadhousehold]

extensions [ nw ]

households-own [
  tension
  accuser
  faccuser
  faccused
  accused
  health
  countlinkneighbors
  countlinkneighborsprime
  deltalinkneighbors
  dead
  stopsign
  accuseswitch
  linkswitch
  distance-from-other-households ; list of distances of this node from other households
  my-clustering-coefficient   ; the current clustering coefficient of this node
  my-clustering-coefficient-round
]


to startup
  set highlight-string ""
end

to setup
  clear-all
  reset-ticks

  set pollution 0
  set pvar 0
  set totgeneralhealth 0
  set totvictimhealth 0
  set totleaderhealth 0
 set  avggeneralhealth 0
  set avgvictimhealth 0
  set avgleaderhealth 0
  set totgeneralcc 0
  set totvictimcc 0
  set totleadercc 0
  set avggeneralcc 0
  set avgvictimcc 0
  set avgleadercc 0
  set generalcc 0
  set leadercc 0
  set victimcc 0
  set totgenerallinkneighbors 0
  set totvictimlinkneighbors 0
  set totleaderlinkneighbors 0
  set avggenerallinkneighbors 0
  set avgvictimlinkneighbors 0
  set avgleaderlinkneighbors 0
  set avggenerallinkneighbordelta 0
  set avgvictimlinkneighbordelta 0
  set avgleaderlinkneighbordelta 0
  set deadcircles 0
  set deadstars 0
  set deadtriangles 0
  set totcircles 0
  set tottriangles 0
  set totstars 0
  set totsquares 0
  set generaldr 0
  set victimdr 0
  set faccuserdr 0
  set timetoritual 0
  set timetoritualswitch 0
  set ritualtime 0
  set ticktimer 0
  set ticktimerswitch 0
  set timetoleader 0
  set timetoleaderswitch 0

  ; set the global variables
  set infinity 99999
  set highlight-string ""

  ; make the nodes and arrange them in a circle in order by who number
  set-default-shape households "circle"
  create-households numnodes
  layout-circle (sort households) max-pxcor - 1
```

```
  ; Create the initial lattice
  wire-lattice

  ; Fix the color scheme
  ask households [
    set color green
    set tension 0
    set accuser 0
    set accused 0
    set dead 0
    set health 4
    set countlinkneighbors 0
    set countlinkneighborsprime 0
    set stopsign green
  ]
  ask links [ set color gray + 2 ]

  ; Calculate the initial average path length and clustering coefficient
  set average-path-length find-average-path-length
  set clustering-coefficient find-clustering-coefficient

  set average-path-length-of-lattice average-path-length
  set clustering-coefficient-of-lattice clustering-coefficient

  create-wells 1 [setxy 0 0]
  ask wells
  [
    set shape "circle"
    set size 4
    set color blue
    set pollution 0
  ]

end

to-report in-neighborhood? [ hood ]
  report ( member? end1 hood and member? end2 hood )
end


to-report find-clustering-coefficient

  let cc infinity

  ifelse all? households [ count link-neighbors <= 1 ] [
    ; it is undefined
    ; what should this be?
    set cc 0
  ][
    let total 0
    ask households with [ count link-neighbors <= 1 ] [ set my-clustering-coefficient "undefined"
]
    ask households with [ count link-neighbors > 1 ] [
      let hood link-neighbors
      set my-clustering-coefficient (2 * count links with [ in-neighborhood? hood ] /
                                  ((count hood) * (count hood - 1)) )
      ; find the sum for the value at households
      set total total + my-clustering-coefficient
    ]
    ; take the average
    set cc total / count households with [count link-neighbors > 1]
  ]

  report cc
end


to-report find-average-path-length

  let apl 0

  find-path-lengths

  let num-connected-pairs sum [length remove infinity (remove 0 distance-from-other-
households)] of households
  ifelse num-connected-pairs != (count households * (count households - 1)) [
    set apl infinity
  ][
    set apl (sum [sum distance-from-other-households] of households) / (num-connected-pairs)
  ]

  report apl
end


to find-path-lengths
  ask households [
    set distance-from-other-households []
  ]

  let i 0
  let j 0
  let k 0
  let node1 one-of households
  let node2 one-of households
```

```
 let node-count count households
 while [i < node-count] [
  set j 0
  while [ j < node-count ] [
   set node1 household i
   set node2 household j
   ifelse i = j [
    ask node1 [
     set distance-from-other-households lput 0 distance-from-other-households
    ]
   ][
    ifelse [ link-neighbor? node1 ] of node2 [
     ask node1 [
      set distance-from-other-households lput 1 distance-from-other-households
     ]
    ][ ;
     ask node1 [
      set distance-from-other-households lput infinity distance-from-other-households
     ]
    ]
   ]
   set j j + 1
  ]
  set i i + 1
 ]
 set i 0
 set j 0
 let dummy 0
 while [k < node-count] [
  set i 0
  while [i < node-count] [
   set j 0
   while [j < node-count] [
    set dummy ( (item k [distance-from-other-households] of household i) +
          (item j [distance-from-other-households] of household k))
    if dummy < (item j [distance-from-other-households] of household i) [
     ask household i [
      set distance-from-other-households replace-item j distance-from-other-households
dummy
     ]
    ]
    set j j + 1
   ]
   set i i + 1
  ]
  set k k + 1
 ]
```

```
end

to wire-lattice
  ; iterate over the households
  let n 0
  while [ n < count households ] [
    ; make edges with the next two neighbors
    ; this makes a lattice with average degree of 4
    make-edge household n
              household ((n + 1) mod count households)
              "default"
    ; Make the neighbor's neighbor links curved
    make-edge household n
              household ((n + 2) mod count households)
              "curve"
    set n n + 1
  ]


  ask link 0 (count households - 2) [ set shape "curve-a" ]
  ask link 1 (count households - 1) [ set shape "curve-a" ]
end

to make-edge [ node-A node-B the-shape ]
  ask node-A [
    create-link-with node-B  [
      set shape the-shape
    ]
  ]
end


to highlight
  ask households [ set color gray + 2 ]
  ask links   [ set color gray + 2 ]

  if mouse-inside? [ do-highlight ]


  display
end

to do-highlight
  let min-d min [ distancexy mouse-xcor mouse-ycor ] of households
  let node one-of households with [count link-neighbors > 0 and distancexy mouse-xcor mouse-
ycor = min-d]

  if node != nobody [
    ask node [
      set color white
      let pairs (length remove infinity distance-from-other-households)
      let my-apl (sum remove infinity distance-from-other-households) / pairs

      let coefficient-description ifelse-value my-clustering-coefficient = "undefined"
        ["undefined for single-link"]
        [precision my-clustering-coefficient 3]
      set highlight-string (word "clustering coefficient = " coefficient-description
        " and avg path length = " precision my-apl 3
        " (for " pairs " households )")
    ]

    let neighbor-nodes [ link-neighbors ] of node
    let direct-links [ my-links ] of node

    ask neighbor-nodes [
      set color orange
      ask my-links [
        ifelse (end1 = node or end2 = node)
          [ set color orange ]
          [ if (member? end1 neighbor-nodes and member? end2 neighbor-nodes) [ set color
yellow ]
        ]
      ]
    ]
  ]
end

to go
  if any? households with [shape = "square" and dead = 0]
  [set ticktimerswitch 1]

  if ticktimerswitch = 1
  [set ticktimer ticktimer + 1]

  if ticktimer > 1999
  [
    if datacollector?
  [
  file-open "scapegoat.txt"
    file-print (word numnodes ", " friendliness " , "skepticism " , "pctvictims " , "
avggeneralhealth " , " avgleaderhealth " , " avgvictimhealth " , " avggenerallinkneighbors " , "
avgvictimlinkneighbors " , " avgleaderlinkneighbors " , " avggenerallinkneighbordelta " , "
avgvictimlinkneighbordelta " , " avgleaderlinkneighbordelta " ," avggeneralcc " , " avgleadercc "
, " avgvictimcc " , " generaldr " , " faccuserdr " , " victimdr " , "timetoritual" , "ritualtime ",
"timetoleader)
    file-close
    ]
     setup
  ]

    tick

  ask households
  [
    set stopsign green
    set accuseswitch green
    set linkswitch green
  ]




  if any? households with [accused = 1]
  [
    ask households with [dead = 0]
    [
      set tension 0
      set accused 0
      set faccuser 0
      set faccused 0
    ]
  ]

  ask households with [dead = 0 and color != blue]
  [
    set accuser 0
    set accused 0
    set faccuser 0
    set faccused 0
  ]

  ask households with [dead = 0 and color != blue]
  [
    if tension < 0
    [set tension 0]
    if tension > 3
    [set tension 3]

    if tension = 0
    [set color green]
    if tension = 1
    [set color yellow]
    if tension = 2
    [set color orange]
    if tension = 3
    [set color red]


  ]


  if random 100 > 93
  [
    ask one-of links
    [die]
  ]

  if random 100 > 93
  [
    ask one-of households with [dead = 0 and color != blue]
    [
      create-link-with one-of other households with [dead = 0] who-are-not link-neighbors
    ]
  ]

  if random 100 > 90 and count households with [dead = 1] > 0
  [
    ask one-of households with [dead = 1]
    [
      set dead 0
      set accuser 0
      set accused 0
      set faccuser 0
      set faccused 0
      set tension 0
      set health 3
      set shape "circle"
      set color green
      set size 1
      create-link-with min-one-of other households who-are-not link-neighbors [distance myself]
      create-link-with min-one-of other households who-are-not link-neighbors [distance myself]
      create-link-with min-one-of other households who-are-not link-neighbors [distance myself]
      create-link-with min-one-of other households who-are-not link-neighbors [distance myself]
      set countlinkneighborsprime 4
    ]
  ]




  ask households with [dead = 0]
```

```
[
  if random 100 = 1
  [
    set tension tension + 1
    set stopsign red
  ]
]




ask households with [dead = 0]
[
  if health < 4
  [set health health + 0.1]
  if tension > 0
  [
    if pollution = 1
    [
      set tension tension - random 1
      set stopsign red
    ]
    if pollution = 2
    [
      set tension tension + random 1
      set stopsign red
    ]
    if pollution = 3
    [
      set tension tension + 1
      set stopsign red
    ]
  ]

  if tension > 2
  [
    ask link-neighbors
    [
    if stopsign = green
    [
        if random 100 > 50
        [
      set tension tension + 1
      set stopsign red
        ]
    ]
    ]
  ]
]

ask households with [dead = 0]
[
  if all? link-neighbors [color = red]
  [
    if random 100 > 74
    [
      set health health - 0.5
    ]
  ]
]

if any? households with [shape = "square" and color != blue]
[
  if random 100 > 97
  [
    ask one-of households with [shape = "square" and color != blue]
    [
      set shape "circle"
      set size 1
    ]
  ]
]


ask households
[
  if health > 4
  [
    set health 4
  ]
  if health < 1
  [
    set dead 1
    set color black
    set size 0
    ask my-links [die]
    if shape = "circle"
  [
      set deadcircles deadcircles + 1
]
    if shape = "star"
  [
      set deadstars deadstars + 1
]
    if shape = "triangle"
    [
      set deadtriangles deadtriangles + 1
  ]
    ]
  ]





ask households
  [
  if scapegoat?
    [

     if count households with [shape = "star" and dead = 0] > count households with [dead = 0] /
10 and any? households with [shape = "square" and dead = 0] and not any? households with
[color = blue]
        [
        ask one-of households with [dead = 0 and shape = "square"]
        [
          set accuser 1
                  set shape "square"
                  set color blue
                  set size 3
                ask one-of households with [shape = "star" and dead = 0]
                [
                  set accused 1
                  set size 3
                  set color blue
                 set health 0
              if not datacollector?
              [
                wait 1
              ]
                ask other households with [shape != "square" and dead = 0 and color != blue]
                [
                  set shape "circle"
                  set color green
                  set size 1
                  set accuser 0
                  set accused 0
                  set tension 0
                    if nw:distance-to one-of other households with [dead = 0 and shape = "square"] !=
false
                    [
                    if random 100 > ((65 + friendliness + (nw:distance-to min-one-of other households
with [dead = 0 and shape = "square"] [distance myself] * 2)))
```

```
              [
              create-link-with one-of other households with [shape = "square" and dead = 0]
              ]
              ]

           ]
          ask households
          [set accuseswitch red]
           ]
          ]
    ]



    if accuseswitch = green
    [
      if any? households with [shape = "square" and color = blue]
      [
        if count households with [color = red] > (.95 * (count households with [dead = 0] - count
households with [color = blue]))
        [
        ifelse any? households with [shape = "star" and dead = 0]
        [
        ask one-of households with [shape = "star" and dead = 0]
        [
             set health 0
            set accused 1
            ask other households with [shape != "square" and dead = 0 and color != blue]
            [
              set shape "circle"
              set color green
              set size 1
              set accuser 0
              set accused 0
              set tension 0
                if nw:distance-to one-of other households with [dead = 0 and shape = "square"] !=
false
                [
                if random 100 > ((50 + friendliness + (nw:distance-to min-one-of other households
with [dead = 0 and shape = "square"] [distance myself] * 2)))
                [
                create-link-with one-of other households with [shape = "square" and dead = 0]
                ]
            ]
              ]
            ask households
                [set accuseswitch red]
          ]
          ]

        [
            ask households with [color = blue]
            [
              set shape "circle"
              set color green
              set size 1
              ask households
              [set accuseswitch red]
            ]
          ]
        ]
      ]
    ]





    if accuseswitch = green and not any? households with [color = blue]
    [
      ifelse count households with [shape = "square" and dead = 0] < (count households with
[dead = 0] / 20)
        [

  ask one-of households with [dead = 0 and color != blue]
  [
    if tension = 3 and all? link-neighbors [color = red]
    [


              while [accuseswitch = green]
              [
          if random 100 > 74 and accuseswitch = green
        [
            if any? other households with [dead = 0 and count link-neighbors < 2]
          [
                set faccuser 1
                set shape "triangle"
                set color white - 2
                set size 2
                set health health - random 5
                ask one-of other households with [dead = 0 and count link-neighbors < 2]
        [
              set faccused 1
                      if color != blue
                      [
              set color white - 2
              set shape "x"
              set size 2
                 ask households
                [
              set accuseswitch red
                ]
                ]
           ]
    set stopsign red
              stop

        ]
        ]

          if random 100 > skepticism and accuseswitch = green
        [
           if any? households with [dead = 0 and shape = "x" and nw:distance-to min-one-of
households [distance myself] != false]
          [
             if any? households with [dead = 0 and shape = "x" and nw:distance-to min-one-of
households [distance myself] > 2 and nw:distance-to min-one-of households [distance myself]
< 6]
          [
             set faccuser 1
             set shape "triangle"
             set color white - 2
             set size 2
             set health health - random 5
             ask one-of households with [dead = 0 and shape = "x" and nw:distance-to min-one-
of households [distance myself] > 2 and nw:distance-to min-one-of households [distance
myself] < 6]
        [
            set faccused 1
                 if color != blue
                 [
              set color white - 2
              set shape "x"
              set size 2
                 ]
                 ask households
                [
              set accuseswitch red
                ]
           ]
               set stopsign red
               stop
            ]
    ]
    ]

    if random 100 > skepticism and accuseswitch = green
    [
      if any? households with [dead = 0 and nw:distance-to min-one-of households [distance
myself] != false]
     [
       if any? households with [dead = 0 and nw:distance-to min-one-of households
[distance myself] > 2 and nw:distance-to min-one-of households [distance myself] < 6]
     [
        set faccuser 1
        set shape "triangle"
        set color white - 2
        set size 2
        set health health - random 5
        ask one-of households with [dead = 0 and nw:distance-to min-one-of households
[distance myself] > 2 and nw:distance-to min-one-of households [distance myself] < 6]
    [
       set faccused 1
            if color != blue
            [
         set color white - 2
         set shape "x"
         set size 2
            ]
            ask households
           [
         set accuseswitch red
           ]
      ]
        set stopsign red
          stop
       ]
    ]
    ]

     if random 100 > skepticism and accuseswitch = green
    [
        set faccuser 1
        set shape "triangle"
        set color white - 2
        set size 2
        set health health - random 5
        ask one-of households with [dead = 0 and shape != "square"]
    [
       set faccused 1
```

```
                if color != blue
                [
              set color white - 2
              set shape "x"
              set size 2
                ]
                ask households
                  [
              set accuseswitch red
                  ]
        ]
            set stopsign red
            stop
          ]




if random 100 > 50 and accuseswitch = green
          [
            if any? households with [dead = 0 and count link-neighbors < 2]
            [
              set accuser 1
              set shape "square"
              set color white
              set size 2
              ask one-of households with [dead = 0 and count link-neighbors < 2]
          [
           set accused 1
                  if color != blue
                  [
          set shape "star"
           set color white
                set size 2
                  ]
            set health health - random 5
            ask households
                  [
                set accuseswitch red
                  ]
                ]
              set stopsign red
                stop
     ]
           ]

          if random 100 > 50 and accuseswitch = green
          [
          if any? households with [dead = 0 and shape = "x" and nw:distance-to min-one-of
households [distance myself] != false]
          [
            if any? households with [dead = 0 and shape = "x" and nw:distance-to min-one-of
households [distance myself] > 2 and nw:distance-to min-one-of households [distance myself]
< 6]
         [
              set accuser 1
              set shape "square"
              set color white
              set size 2
              ask one-of households with [dead = 0 and shape = "x" and nw:distance-to min-one-
of households [distance myself] > 2 and nw:distance-to min-one-of households [distance
myself] < 6]
        [
         set accused 1
                  if color != blue
                  [
         set shape "star"
         set color white
               set size 2
                  ]
          set health health - random 5
          ask households
                 [
               set accuseswitch red
                 ]
                 ]
              set stopsign red
                 stop
        ]
   ]
         ]


        if random 100 > 50 and accuseswitch = green
        [
          if any? households with [dead = 0 and nw:distance-to min-one-of households [distance
myself] != false]
          [
            if any? households with [dead = 0 and nw:distance-to min-one-of households
[distance myself] > 2 and nw:distance-to min-one-of households [distance myself] < 6]
         [
              set accuser 1
              set shape "square"
              set color white
              set size 2
              ask one-of households with [dead = 0 and nw:distance-to min-one-of households
[distance myself] > 2 and nw:distance-to min-one-of households [distance myself] < 6]
        [
          set accused 1
                      if color != blue
                      [
         set shape "star"
          set color white
                set size 2
                      ]
           set health health - random 5
            ask households
                    [
                set accuseswitch red
                    ]
              set stopsign red
                 stop
        ]
   ]
          ]
               ]



        if random 100 > 50 and accuseswitch = green
        [
          set accuser 1
               set shape "square"
               set color white
            set size 2
            ask one-of households with [dead = 0 and shape != "square"]
          [
            set accused 1
         set shape "star"
          set color white
             set size 2
           set health health - random 5
           ask households
                    [
                set accuseswitch red
                    ]
          ]
          set stopsign red
               stop
        ]

        ]
           ]
          ]
     ]
```

```
    [

      ask one-of households with [dead = 0 and shape = "square" and color != blue]
        [

          if tension = 3 and all? link-neighbors [color = red]
    [
           while [accuseswitch = green]
           [

        if random 100 > ((skepticism / 100) * 10 + 90) and accuseswitch = green
          [

             if any? households with [dead = 0 and nw:distance-to min-one-of households
[distance myself] != false]
         [
              if any? households with [dead = 0 and nw:distance-to min-one-of households
[distance myself] > 2 and nw:distance-to min-one-of households [distance myself] < 6]
         [
                 set faccuser 1
          set shape "triangle"
              set color white - 2
                set size 2
              set health health - random 5
               ask min-one-of households with [dead = 0 and nw:distance-to one-of households
with [faccuser = 1] > 2 and nw:distance-to one-of households with [accuser = 1] < 6] [distance
myself]
     [
           set faccused 1
                 if color != blue
                 [
                set shape "x"
                set color white - 2
                 set size 2
                 ]
                 ask households
                [
              set accuseswitch red
                ]
        ]
               set stopsign red
               stop
     ]
        ]

       ]
```

```
if random 100 > 50 and accuseswitch = green
       [
         if any? households with [dead = 0 and count link-neighbors < 2]
      [
          set accuser 1
          set color white
          set size 2
          ask one-of households with [dead = 0 and count link-neighbors < 2]
          [
      set accused 1
              if color != blue
              [
           set color white
           set shape "star"
               set size 2
              ]
           set health health - random 5
       ask households
             [
           set accuseswitch red
             ]
          ]
           set stopsign red
           stop
     ]
  ]


      if random 100 > 50 and accuseswitch = green
      [
         if any? households with [dead = 0 and nw:distance-to min-one-of households
[distance myself] != false] and count households with [shape = "x"] > 0
      [
          if any? households with [dead = 0 and shape = "x" and nw:distance-to min-one-of
households [distance myself] > 2 and nw:distance-to min-one-of households [distance myself]
< 6]
          [
          set accuser 1
          set color white
          set size 2
          ask one-of households with [dead = 0 and shape = "x" and nw:distance-to min-one-
of households [distance myself] > 2 and nw:distance-to min-one-of households [distance
myself] < 6]
          [
      set accused 1
               if color != blue
```

```
                 [
             set color white
             set shape "star"
                 ]
                  set size 2
             set health health - random 5
         ask households
               [
             set accuseswitch red
               ]
            ]
              set stopsign red
              stop
        ]
        ]
         ]



         if random 100 > 50 and accuseswitch = green
         [
           if any? households with [dead = 0 and nw:distance-to min-one-of households
[distance myself] != false]
         [
            if any? households with [dead = 0 and nw:distance-to min-one-of households
[distance myself] > 2 and nw:distance-to min-one-of households [distance myself] < 6]
       [
           set accuser 1
            set color white
             set size 2
             ask one-of households with [dead = 0 and nw:distance-to min-one-of households
with [accuser = 1] [distance myself] > 2 and nw:distance-to min-one-of households with
[accuser = 1] [distance myself] < 6]
        [
           set accused 1
                 if color != blue
                 [
              set color white
              set shape "star"
              set size 2
                 ]
              set health health - random 5
         ask households
               [
             set accuseswitch red
               ]
             ]
              set stopsign red
              stop
        ]
        ]
          ]


          if random 100 > 50 and accuseswitch = green
        [
             set accuser 1
               set color white
                 set size 2
             ask one-of households with [dead = 0 and shape != "square"]
        [
             set accused 1
                     if color != blue
                    [
                   set color white
                   set shape "star"
                   set size 2
                    ]
                   set health health - random 5
           ask households
                    [
                 set accuseswitch red
                    ]
                ]
               set stopsign red
               stop
        ]
             ]
           ]
      ]
    ]
]
]
]




if not any? households with [color = blue]
[

if any? households with [dead = 0 and count link-neighbors < 2]
[
        if random 100 > (80 + friendliness)
        [
         ask households with [dead = 0 and count link-neighbors < 2 and color != blue]
        [
      ifelse any? other households with [dead = 0 and (shape = "star" or shape = "x") and color
!= blue]
  [
         create-link-with one-of other households with [dead = 0 and (shape = "star" or shape =
"x") and color != blue]
         ]
      [
     if random 100 > (90 + friendliness / 2)
     [
      ask households with [dead = 0 and count link-neighbors < 2 and color != blue]
      [
     create-link-with one-of other households with [dead = 0 and color != blue]
      ]
   ]
   ]
    ]
   ]
   ]


  if any? households with [dead = 0 and shape = "square" and color != blue] and any?
households with [dead = 0 and shape = "star" and color != blue]
   [
     if any? households with [dead = 0 and shape = "star" and color != blue and nw:distance-to
min-one-of other households with [shape = "square" and color != blue] [distance myself] !=
false]
   [
      ask one-of households with [dead = 0 and shape = "star" and color != blue and
nw:distance-to min-one-of other households with [shape = "square"] [distance myself] !=
false]
  [
       ifelse nw:distance-to min-one-of other households with [dead = 0 and shape = "square"
and color != blue] [distance myself] != false
       [
         if nw:distance-to min-one-of other households with [dead = 0 and shape = "square" and
color != blue] [distance myself] < 6
           [
         if (random  100 > (80 + friendliness) + nw:distance-to min-one-of other households with
[dead = 0 and shape = "square"] [distance myself]) and linkswitch = green
       [
         create-link-with one-of other households with [dead = 0 and shape = "square"]
         set linkswitch red
       ]
       if random 100 > ((80 + (friendliness / 2)) + nw:distance-to min-one-of other households
with [dead = 0 and shape = "square"] [distance myself]) and linkswitch = green and any? other
households with [shape = "square" and count link-neighbors > 0]
       [
            if any? other households with [color != blue and nw:distance-to one-of households
with [dead = 0 and shape = "square"] != false]
              [
                if any? other households with [color != blue and nw:distance-to min-one-of
households with [dead = 0 and shape = "square"] [distance myself] = 1]
                [
                  create-link-with min-one-of other households with [nw:distance-to min-one-of
households with [dead = 0 and shape = "square"] [distance myself] = 1] [distance myself]
          set linkswitch red
        ]
        ]
        ]
  ]
  ]
        []
      ]
  ]
    ]

  if any? households with [dead = 0 and shape = "star" and color != blue] and any? households
with [dead = 0 and shape = "square" and color != blue]
  [
  if any? households with [dead = 0 and shape = "star" and color != blue and nw:distance-to
min-one-of households with [shape = "square"] [distance myself] != false]
  [
    ask one-of households with [dead = 0 and shape = "star" and color != blue and nw:distance-
to min-one-of households with [shape = "square"] [distance myself] != false]
    [
    if nw:distance-to min-one-of households with [dead = 0 and shape = "square"] [distance
myself] > 2 and nw:distance-to min-one-of households with [dead = 0 and shape = "square"]
[distance myself] < 6
      [
        if count link-neighbors > 0
        [
          if any? link-neighbors with [nw:distance-to min-one-of households with [dead = 0 and
shape = "square"] [distance myself] != false]
          [
            ask link-with min-one-of link-neighbors [nw:distance-to min-one-of households with
[dead = 0 and shape = "square"] [distance myself]]
        [die]
      ]
        ]
      ]
        ]
    ]
    ]

  if any? households with [shape = "square" and dead = 0]
  [
    if any? households with [dead = 0 and color != blue and nw:distance-to min-one-of
households with [dead = 0 and shape = "square"] [distance myself] != false]
  [
     ask households with [dead = 0 and color != blue and nw:distance-to min-one-of households
with [dead = 0 and shape = "square"] [distance myself] != false]
    [
     if nw:distance-to one-of households with [dead = 0 and shape = "square"] = 5
     [
        if count link-neighbors with [dead = 0 and shape = "square"] > 0
        [
          ask link-with one-of link-neighbors with [dead = 0 and shape = "square"]
       [die]
     ]
        ]
    ]
  ]
  ]
  ]

  set tothouseholds tothouseholds + count households with [dead = 0]
set totcircles totcircles + count households with [shape = "circle" and dead = 0]
  set totstars totstars + count households with [shape = "star" and dead = 0]
  set tottriangles tottriangles + count households with [shape = "triangle" and dead = 0]
  set totsquares totsquares + count households with [shape = "square" and dead = 0]

  set totgeneralhealth totgeneralhealth + sum [health] of households with [shape = "circle" and
dead = 0]
    set avggeneralhealth totgeneralhealth / totcircles
  if count households with [shape = "star"] > 0
  [
  set totvictimhealth totvictimhealth + sum [health] of households with [shape = "star" and
dead = 0]
  set avgvictimhealth totvictimhealth / totstars
  ]
  if count households with [shape = "square"] > 0
  [
  set totleaderhealth totleaderhealth + sum [health] of households with [shape = "square" and
dead = 0]
  set avgleaderhealth totleaderhealth / totsquares
  ]



  set pvar random 100
  if pvar < 25 and pollution > 0
  [set pollution pollution - 1]
  if pvar > 75 and pollution < 3
  [set pollution pollution + 1]
```

```
if pollution = 0
[
 ask wells [die]
create-wells 1
[setxy 0 0
   set shape "circle"
 set size 4
 set color blue
]
]

if pollution = 1
   [
ask wells [die]
create-wells 1
[setxy 0 0
 set size 4
 set color green
      set shape "circle"
]
   ]

if pollution = 2
   [
     ask wells [die]
create-wells 1
[setxy 0 0
      set shape "circle"
 set size 4
 set color yellow
]
   ]

if pollution = 3
   [
     ask wells [die]
create-wells 1
[setxy 0 0
      set shape "circle"
 set size 4
 set color brown
]
   ]


ask households with [ count link-neighbors <= 1 ] [ set my-clustering-coefficient 0 ]
 ifelse any? households with [ count link-neighbors > 1 ]
 [
   ask households with [ count link-neighbors > 1 ]
 [
     let hood link-neighbors
     set my-clustering-coefficient (2 * count links with [ in-neighborhood? hood ] /
                                   ((count hood) * (count hood - 1)) )
 ]
 ]
 []

 set generalcc mean [my-clustering-coefficient] of households with [shape = "circle"]

 ifelse count households with [shape = "square"] < 1
 [
   set leadercc 0
 ]
 [
   set leadercc mean [my-clustering-coefficient] of households with [shape = "square"]
 ]

 ifelse count households with [shape = "star"] < 1
 [
   set victimcc 0
 ]
 [
   set victimcc mean [my-clustering-coefficient] of households with [shape = "star"]
 ]

 ask households
 [
 if my-clustering-coefficient < 0.1
 [
   set my-clustering-coefficient-round 0
 ]
if my-clustering-coefficient > 0.1 and my-clustering-coefficient <= 0.2
 [
   set my-clustering-coefficient-round 1
 ]
 if my-clustering-coefficient > 0.2 and my-clustering-coefficient <= 0.3
 [
   set my-clustering-coefficient-round 2
 ]
 if my-clustering-coefficient > 0.3 and my-clustering-coefficient <= 0.4
 [
   set my-clustering-coefficient-round 3
 ]
 if my-clustering-coefficient > 0.4 and my-clustering-coefficient <= 0.5
 [
   set my-clustering-coefficient-round 4
 ]
 if my-clustering-coefficient > 0.5 and my-clustering-coefficient <= 0.6
 [
   set my-clustering-coefficient-round 5
 ]
 if my-clustering-coefficient > 0.6
 [
   set my-clustering-coefficient-round 6
 ]
 ]

 set totgeneralcc totgeneralcc + (generalcc * count households with [shape = "circle" and dead
= 0])
   set avggeneralcc totgeneralcc / totcircles
 if count households with [shape = "star"] > 0
 [
   set totvictimcc totvictimcc + (victimcc * count households with [shape = "star" and dead =
0])
 set avgvictimcc totvictimcc / totstars
 ]
 if count households with [shape = "square"] > 0
 [
   set totleadercc totleadercc + (leadercc * count households with [shape = "square" and dead
= 0])
 set avgleadercc totleadercc / totsquares
 ]

 ask households with [dead = 0]
 [
   set countlinkneighbors countlinkneighbors + count link-neighbors
   set deltalinkneighbors countlinkneighbors - countlinkneighborsprime
   set countlinkneighborsprime countlinkneighbors
 ]

 set totgenerallinkneighbors totgenerallinkneighbors + mean [countlinkneighbors] of
households with [shape = "circle" and dead = 0]
   set avggenerallinkneighbors totgenerallinkneighbors / totcircles
 if count households with [shape = "star" and dead = 0] > 0
 [
   set totvictimlinkneighbors totvictimlinkneighbors + mean [countlinkneighbors] of
households with [shape = "star" and dead = 0]
 set avgvictimlinkneighbors totvictimlinkneighbors / totstars
 ]
 if count households with [shape = "square" and dead = 0] > 0
 [
   set totleaderlinkneighbors totleaderlinkneighbors + mean [countlinkneighbors] of
households with [shape = "circle" and dead = 0]
```

```
  set avgleaderlinkneighbors totleaderlinkneighbors / totsquares
  ]

  set avggenerallinkneighbordelta mean [deltalinkneighbors] of households with [shape =
"circle" and dead = 0]
  if any? households with [shape = "star" and dead = 0]
 [
  set avgvictimlinkneighbordelta mean [deltalinkneighbors] of households with [shape = "star"
and dead = 0]
  ]
  if any? households with [shape = "square" and dead = 0]
  [
  set avgleaderlinkneighbordelta mean [deltalinkneighbors] of households with [shape =
"square" and dead = 0]
  ]

  set pctvictims (totstars / tothouseholds) * 100

  if any? households with [color = blue]
  [
    set timetoritualswitch 1
    set ritualtime ritualtime + 1
  ]

  if not any? households with [color = blue] and timetoritualswitch = 0
  [set timetoritual timetoritual + 1]

  if any? households with [shape = "square" and dead = 0]
  [set timetoritualswitch 1]

  if timetoritualswitch = 0
  [set timetoritual timetoritual + 1]

  set generaldr (deadcircles / totcircles)
  if totstars > 0
  [set victimdr (deadstars / totstars)]
  if tottriangles > 0
  [set faccuserdr (deadtriangles / tottriangles)]



end
```