```
extensions [nw table]
links-own [ weight ]
turtles-own [
  dict;dictionary with shortest path to every node
  insured?
  checked?
  payoff
  cost-of-link-with-other-turtles ;;
  distance-from-other-turtles
  indirpayoffbefore
  indirpayoffafter
  degree
  ]
globals[
donewithinsured?
infinity
newpayoff1
newpayoff2
nolinkpayoff
nolinkpayoff2
nr1
nr2

]
to setup-shape
  clear-all
  setup-patches
  nw:generate-ring turtles links 10 [ set color red ]
 nw:set-snapshot turtles links
  layout
  set infinity 99999
  ask turtles [
    set indirpayoffbefore 0
    set indirpayoffafter 0
    set payoff 0
    set insured? true
    set checked? false
    set color green
    let node-count count turtles
    let x 0
  ]
  compute-inital-payoff
   nw:set-snapshot turtles links
  reset-ticks
end
to setup-star
  clear-all
  setup-patches
  setup-turtles-star
  reset-ticks
end
to setup
  clear-all
  setup-patches
  setup-turtles
  reset-ticks
end

to setup-turtles-star
  setup-patches
  set-default-shape turtles "circle"
  nw:generate-star turtles links num-nodes
  nw:set-snapshot turtles links
  layout
  set infinity 99999
  ask turtles [
    set indirpayoffbefore 0
    set indirpayoffafter 0
    set payoff 0
    set insured? true
    set checked? false
    set color green
    let node-count count turtles
    let x 0
  ]
  compute-inital-payoff
   nw:set-snapshot turtles links
end
to setup-turtles
  set-default-shape turtles "circle"
  set infinity 99999
  crt num-nodes
  layout-circle turtles max-pxcor - 20
  ask turtles [
    set indirpayoffbefore 0
    set indirpayoffafter 0
    set payoff 0
    set insured? true
    set checked? false
    set color green
    let node-count count turtles
    let x 0
  ]
   nw:set-snapshot turtles links
  ;ask turtles [ set label who set label-color black
end

to compute-inital-payoff
  find-path-lengths
  ask turtles [
    set degree count link-neighbors
    let nr who
    let i 0
    let j 1
    set payoff 0
   foreach distance-from-other-turtles [
     if( ? < 999 )[
         if(? != 0)[
           set payoff (payoff +( (beta / 100) ^ ? ))
         ]
         if( ? = 1)[
             set payoff (payoff - ((insurancelink / 100 )  /
(j)))
             set j j + 1
     ]
     ]
   ]
  ]
end

to setup-patches
ask patches [
  set pcolor white
  ]
end

to go
    add-edge-simpler
    delete
    layout
    tick
end
to delete
  let i 0
  while [i < count turtles ]
  [
    check-delete i
    set i i + 1
    ]
end
to add-edge-simpler
  set newpayoff1 -1
  set newpayoff2 -1
  set nolinkpayoff 0
  set nolinkpayoff2 0
  compute-inital-payoff
  let node1 one-of turtles
  if( node1 = nobody)[
    display
    user-message "ferdig"
    stop
  ]
  set nr1 0
  set nr2 0
  let link? false
  ask node1[
    set nolinkpayoff payoff
    set nr1 who
    let node2 one-of turtles with [not link-neighbor?
node1 and (self != node1) and not checked?]
    ifelse node2 = nobody
    [
      set checked? true
    ]
    [
      ask node2 [set nr2 who
        set nolinkpayoff2 payoff
        ]
      set link? true
    ]
  ]
  if( link?)[
  create-and-check-path nr1 nr2
  check-delete nr1
  check-delete nr2
  ]
end

to setup-indivudal-map
  let j 0
  let c count turtles
  while [j < c][
  ask turtle j[
  let i 0
  set dict table:make
  while [i <= c - 1][
  if j != i[
    table:put dict i nw:path-to turtle i
    ]
  set i i + 1
  ;end while
  ]
  ;end ask
  ]
  set j j + 1
  ;end while
  ]

end

to check-delete[a]
  let i 0
  let opay -1
  let dist []
  ask turtle a[
  set opay payoff
  set dist distance-from-other-turtles
  ]
    foreach dist
    [
     if( ? = 1)[
      ;neighbors
      ;i is the turtle nr
      ask link a i[
       die
      ]
      nw:set-snapshot turtles links
      find-path-lengths
      compute-inital-payoff
      ask turtle a[
       if(payoff < opay)
       [
         ;do not delete link
         create-link-with turtle i [ set weight 2.0 ] ]
       nw:set-snapshot turtles links
       ]
       find-path-lengths
       compute-inital-payoff
      ]
       set i i + 1
     ]

end

to create-and-check-path[a b]
  let temp []
  let nextloop? true
  ;create temporary table of pathes from 0 to 2.
  ask turtle a [set temp nw:path-to turtle b]

  let len length temp
  ask turtle a [ create-link-with turtle b [ set weight
2.0 ] ]
  let nlink link a b
  nw:set-snapshot turtles links
  setup-indivudal-map
  find-path-lengths

  let t []
  let i 0
  let oldneighbor -1

  nw:set-snapshot turtles links
  find-path-lengths
  compute-inital-payoff
  if ( ([payoff] of turtle a ) <  nolinkpayoff or
([payoff] of turtle b ) <  nolinkpayoff2 )[
    ;remove new link, and recreate the old.
    ask link a b[
      die
      nw:set-snapshot turtles links
    ]
    if(oldneighbor != -1)[
      ask turtle a [ create-link-with oldneighbor [ set
weight 2.0 ] ]
      nw:set-snapshot turtles links
    ]
  ]
  find-path-lengths
  compute-inital-payoff
  setup-indivudal-map

end

to layout
  repeat 10 [
    layout-spring (turtles with [any? link-neighbors])
links 0.4 6 1
    display  ;; so we get smooth animation
  ]
end

to find-path-lengths
  ;; reset the distance list
  ask turtles
  [
    set distance-from-other-turtles []
  ]

  let i 0
  let j 0
  let k 0
  let node1 one-of turtles
  let node2 one-of turtles
  let node-count count turtles
  ;; initialize the distance lists
  while [i < node-count]
  [
    set j 0
    while [j < node-count]
    [
      set node1 turtle i
      set node2 turtle j
      ;; zero from a node to itself
      ifelse i = j
      [
        ask node1 [
          set distance-from-other-turtles lput 0
distance-from-other-turtles
        ]
      ]
      [
        ;; 1 from a node to it's neighbor
        ifelse [ link-neighbor? node1 ] of node2
        [
          ask node1 [
          set distance-from-other-turtles lput 1
distance-from-other-turtles
         ]
       ]
       [
         ask node1 [
          set distance-from-other-turtles lput infinity
distance-from-other-turtles
         ]
       ]
     ]
     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-
turtles] of turtle i) +
                 (item j [distance-from-other-turtles] of
turtle k))

       if dummy < (item j [distance-from-other-
turtles] of turtle i)
       [
         ask turtle i [
          set distance-from-other-turtles replace-item
j distance-from-other-turtles dummy

         ]
       ]
       set j j + 1
     ]
     set i i + 1
   ]
   set k k + 1
 ]

end
```