# Strip the physical proximity layer out and base everything on network connections

Currently:

    if count\_neighbors\_same\_group ≥ agent.seg

        agent.mood = true

    else

        agent.mood = false

        move\_agent\_single!(agent, model)

    end

    return

end

move\_agent\_single! Still relies on the presence of a physical space; needs to change to making new links with friends of friends (and then breaking old links with friends? Will also need to be done in agent\_step rather than in model\_step otherwise risks breaking connections which have just been made)

* Run with step!(model,agent\_step!,model\_step!)

## Checking whether a neighbour has a graph link will become moot, since this is currently based on spatial neighbours

    for agent in allagents(model)

        #check whether the agent has a graph edge with its neighbours, and if not add an edge.

        for neighbor in nearby\_agents(agent, model)

            if has\_edge(model.social, neighbor.id, agent.id) == false

                add\_edge!(model.social, neighbor.id, agent.id)

            end

        end

    end

## Build in a mechanism for pruning graph edges

* There are going to be too many graph edges if we run this for any significant number of steps
* Make new network connections then break old ones
  + Select new neighbours from neighbours-of-neighbours or randomly?
  + Make variable containing IDs of current neighbours, then use that to break current connections once new ones are established?

# Eight-friends

Model in which every agent has at least eight friends. Connections with the other group are broken until they drop below eight, then the agent makes a new connection to bring them back up to eight. Eight is chosen because it is the most likely number of neighbours an agent would have in a Schelling segregation physical space grid.