# 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!)

## Graph creation in model initialisation needs to be random; currently based on physical proximity

# intialise the graph adding an edge to spatial neighbors

    for agent in model.agents

        for neighbor in nearby\_agents(agent, model)

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

        end

    end

* Needs random selection of some number of other nodes + creation of edges between them

## Agent\_step! Needs to go replace (model.social, 1) with (model.social, i) in a way that doesn’t break

    neigh = Graphs.neighbors(model.social, 1)

    for i in neigh

        count\_neighbours += 1

        if model[1].group == model[i].group

            count\_neighbors\_same\_group += 1

        end

        print(count\_neighbors\_same\_group)

    end

## 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?