Exercice: Phuc Xa evacuation

Building the model step-by-step

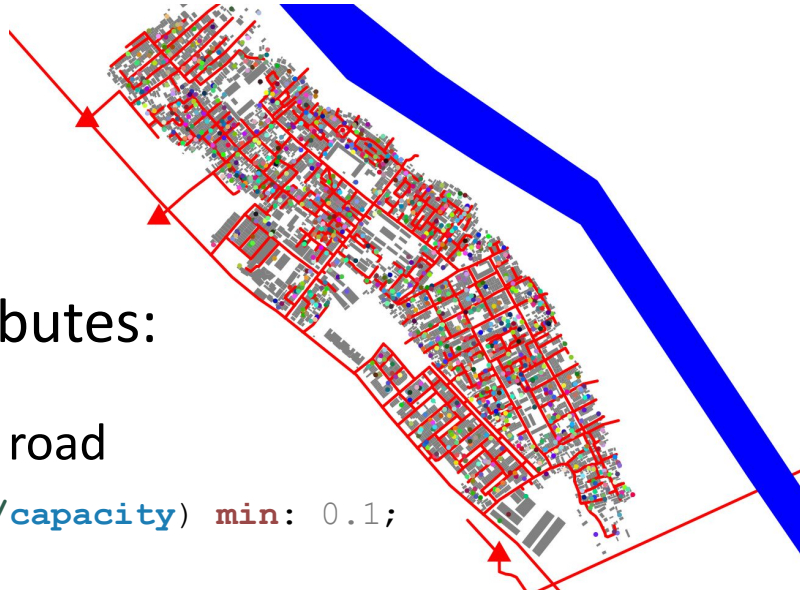
Starting point

- Cleaned up model based on the one from the urban traffic/SIG lesson
- Different place in hanoi
- Import base resources from the drive :

https://t.ly/bz4eT

Step 0: Initialize model with base data

- Set every data as species in the model (`shape_file("...");`)
  - **Buildings** should have a height read from the source file
  - **Red River** should be blue and use the shp shape
  - **Evacuation** has to be represented as a red *triangle(50)*
  - **Inhabitant**
    - 1000 agents
    - starts in a random house
    - use the *moving* skill
    - represented as a random color *circle(5)*
  - **Roads** have to be a usable graph with attributes:
    - Have a *capacity* of *1 + perimeter / 10*
    - *nb_drivers* as the number of inhabitant on the road
    - `float speed_rate <- 1.0 update: exp(-nb_drivers/capacity) min: 0.1;`



Step 0: Initialize model with base data

Step 1: Implementing inhabitants' daily schedule

- Every inhabitant must have a home and a workplace
- 8am-7pm => work, else home

Step 1: solution

- Every inhabitant must have a home and a workplace
  - Add the attributes for the location of workplace and home in the inhabitant species
  - In the initialization of the agents, pick two locations from the building list
- 8am-7pm => work, else home
  - Add a reflex to pick the correct target in function of the time

Step 1: solution

- Every inhabitant must have a home and a workplace
  - Add the attributes for the location of workplace and home in the inhabitant species
  - In the initialization of the agents, pick two locations from the building list
- 8am-7pm => work, else home
  - Add a reflex to pick the correct target in function of the time

```
reflex pick_target when:target = nil {
    if (  current_date.hour#h + current_date.minute#minute >= 8#h
        and current_date.hour#h + current_date.minute#minute <= 17#h
    ){
        target <- work;
    }
    else{
        target <- home;
    }
}
```

Step 2: Trigger evacuation

- Load all the evacuation points from the data
- On flooding date, give every inhabitant an evacuation point (the closest)
    - At 8:30 of the same day
- Move to that point through the roads
- Bonus: stop the simulation once everybody reached an evacuation point

Step 2: Solution

- On flooding date, give every inhabitant an evacuation point (the closest)
- Define a flooding date as global variable

```
date flooding_date <- date([1980,1,2,8,30,0]);
```

- Create a reflex to find the nearest evacuation point

```
reflex pick_target when:target = nil {
  //If we are after the start of the flooding we flee
  if (current_date > flooding_date){
    target <- (evacuation closest_to location).location;
  }
  //else it's a normal day of work
  else{
    if (    current_date.hour#h + current_date.minute#minute >= 8#h
        and current_date.hour#h + current_date.minute#minute <= 17#h
    ){
      target <- work;
    }
    else{
      target <- home;
    }
  }
}
```

Step 2: Solution

- Move to that point through the roads
- Bonus: stop the simulation once everybody reached an evacuation point

```
reflex all_evacuated {
//If at least one inhabitant is not located at an evacuation point
//we continue, else we stop the simulation
  loop i over:inhabitant{
    if evacuation none_matches (i.location = each.location){
      return;
    }
  }
  do pause;
}
```

Optional step for fast learners: creating a river agent

- Load from the shapefile into an agent
- Get it growing every step starting from the flooding date
- Bonus: overlapped roads are unusable, overlapped people die

Optional step for fast learners: solution

- Load from the shapefile into an agent
- Get it growing every step starting from the flooding date
- Bonus: overlapped roads are unusable, overlapped people die

Step 3: Statistics

- Create a new display to show stats
- Plot the number of evacuees
- Parametrize the simulation to show the impact of different parameters:
  - Population
  - Hour of the flood
  - Evacuation point picking strategy (closest vs random vs other ?)
  - Different ways to propagate the news of the flood (instant vs close to the river or to someone who is fleeing)
- For models with dynamic river:
  - Plot the number of deaths, number of closed roads
  - Parametrize the speed of flooding

Step 3: solution

- Create a new display to show stats
- Plot the number of evacuees

```
int nb_evacuee <- 0 update:current_date > flooding_date ? inhabitant count (each.location = location) : 0;
```

```
display stats refresh:every(10#cycles){
  chart "Evacuation status" type:series {
    loop ev over:evacuation{
      data ev.name value:ev.nb_evacuee style:dot;

    }
  }
}
```

- Parametrize the simulation to show the impact of different parameters:
  - Population
  - Hour of the flood
  - Evacuation point picking strategy (closest vs random vs other ?)
  - Different ways to propagate the news of the flood (instant vs close to the river or to someone who is fleeing)
- For models with dynamic river:
  - Plot the number of deaths, number of closed roads
  - Parametrize the speed of flooding

To go further

- Take into account multiple ways of evacuating (walking, motorbike, car), it will change the agent's speed but also the roads capability to "carry" agents
- Run multiple simulations with different parameters at the same time and plot the result into one chart