Documentation on the scenario generator

Alicia Vanhulle

July 2019

Part I
Introduction

The repository evacsim contains a scenario generator for evacuation planning in the event of wildfire.

The generation of instances is in two steps. First it generates the road network. Then, it simulates a fire propagation and determines evacuation routes.

The road network is represented by a graph and the evacuation routes by a tree.

The simulation information is written in two output files (.full and .evac).

In this project, time is discretized by minute, and the indicated lengts have to be multiplied by 50m.

Part II
Installation

Requirements: numpy, matplotlib

- install networkx

```
pip install [--user] networkx
```

- install quadtree

```
python setup.py install [--prefix "some place where PYTHONPATH
                 points to and where you have write permissions"]
```

Part III
Instances

1 Generation of instances

1.1 Generate automatically a data set

Run the file gen_dataset :

```
python gen_dataset.py
```

This data set contains instances generated from three road networks (sparse, medium, dense).

1.2 Generate personalised instances

The generation of instances is in two steps. First, generate the road network. Then, simulate a fire and the associated evacuation routes.

1.2.1 Generate road network:

The following command allows to generate the road network. It will create two files : FILENAME.graph and FILENAME.pos

```
python generator.py --road
                    [--size SIZE] [--limit LIMIT] [--grid GRID]
                    [--printroad]
                    FILENAME
```

–size SIZE: Size of the land area (unit: 50m, ¡int¿, default=1000 (ie 50 km))
–limit LIMIT: (Roughly) the number of intersections in the road network (¡int¿, default=800)
–grid GRID: Density of the primary road grid (¡int¿, default=4)
–printroad: Show the road network (legend: black: secondary roads, green: primary roads, blue: highways)

1.2.2 Simulate fire and evacuation routes

The following command allows to simulate a fire and determine evacuation routes. The simulation information will be written in two files : FILENAME.full and FILENAME.evac (in the folder data).

```
python generator.py --evacuation
                    [--speed SPEED]
                    [--datefactor DATEFACTOR]
                    [--flowfactor FLOWFACTOR] [--increaseflow INCREASEFLOW]
                    [--intensity INTENSITY] [--firepace FIREPACE]
                    [--step STEP] [--num_iter NUM_ITER]
                    [--num_evacuations NUM_EVACUATIONS] [--seed SEED]
                    [--printfire] [--printevac]
                    [--tofile] [--writetree]
                    FILENAME
```

–speed SPEED: Speed of evacuation (unit: 50m/minute, ¡int¿, default=25 (ie 20m/s))
–datefactor DATEFACTOR: Priority given to safe edges in evacuation (¡int¿, default=10)
–flowfactor FLOWFACTOR: Priority given to faster edges in evacuation (¡int¿, default=5)
–increaseflow INCREASEFLOW: Multiplier applied to flow rates (¡float¿, default=1)
–intensity INTENSITY: Intensity of the fire (¡float¿in the interval [0,1], default=.5)
–step STEP: Granularity of the fire areas (unit: 50m, ¡int¿, default=10 (ie 500m)
–num_iter NUM_ITE: Number of wildfire expansion steps (¡int¿ default=100)
–firepace FIREPACE: Time between wildfire expansion (unit: minute, ¡int¿, default=5)
–num_evacuations NUM_EVACUATIONS: Number of evacuation zones (¡int¿, default=10)
–seed SEED: Random seed for the start of fire
–printfire: show the fire simulation
–printevac: show the evac simulation
–tofile: print the graph in a readable format (in the file .full)
–writetree: write the tree in the file .full (no effect without the parameter tofile)

1.2.3 Examples

- Simple example with display

  ```
  python generator.py --road --printroad test
  python generator.py --evacuation --printfire --printevac test
  ```

- Small instance
  In order to have a smaller instance, you can reduce the number of nodes thanks to the limit parameter:

  ```
  python generator.py --road --limit 100 small
  ```

  Comment: If limit¡100, the road network will be composed only of two types of roads : secondary and primary roads (highways cannot be created with a low number of nodes)

  Then, you can generate a plan evacuation for only few nodes :

  ```
  python generator.py --evacuation --num_evacuations 3 small
  ```

- Instance with later deadlines

In order to have later deadlines, the fire propagation can be slowed down. To that purpose, you can reduce the value of step and increase the one of firepace. Nevertheless, it is possible that not enough nodes burn to create an instance. In this case, you should increase num_iter (the inconvenient is a longer execution time).
Example with the dense road network :

```
./generator.py --evacuation --firepace 5 --step 3 --num_iter 150
                            --tofile --writetree dense
```

2 Output files

2.1 File .full

The output file .full is written in a readable format and it contains the whole simulation information. To obtain this file, do not forget the parameter '–tofile'.

- **Evacuation information**
  Format:
  header: <number of evac nodes> <id of safe node>
  one line per evac node: <id of the node> <population> <max rate> <due date> <k> <v1> ... <vk>

  population : number of persons to evacuate
  max rate : maximum of persons who can be evacuated by minute
  due date: date (in minutes) at the latest the last evacuated group have to quit the evacuation node
  k: number of nodes in the escape route
  v1 ... vk: id of the nodes in the escape route

  Comment: max rate = min(minimal capacity of escape route arcs, population)

- **Reduced evacuation tree** (if parameter '–writetree')
  In the reduced evacuation tree, the arcs which belongs to the same escape routes are concatenated.

  Format
  header: <num edges>
  one line per evac node: <id of the son node> <id of the father node> <length> <capacity>

  Comment: The capacities are decreasing along the branches of the reduced tree. As a result, the max rate of an evacuation node = min(max rate of the first arc of the escape route, population of the evacuation node)

length: time (in minutes) for a group to cross the arc
capacity: maximum number of persons who can arrive in the arc every minute

- **Road network** (arcs of the graph)
  Format:
  header: <num nodes> <num edges>
  one line per evac node: <id of the node 1> <id of the node 2> <due date> <length> <capacity>

  due date: date (in minutes) on which the arc will not be practicable any more because of the fire
  length: time (in minutes) for a group to cross the arc
  capacity: maximum number of persons who can arrive in the arc every minute

  Comment: The road network is composed only of two-way roads. Thus, graph arcs are not directed : the order between node 1 and node 2 do not matter.

2.2 File .evac

```
FORMAT :

n m
population_1 maximum_rate_1 duedate_1
        ...
population_n maximum_rate_n duedate_n
capacity_1 k_1 i_1_1 offset_i_1_1 ... i_k_1 offset_i_k_1
        ...
capacity_m k_m i_1_m offset_i_1_m ... i_k_m offset_i_k_m
```

n is the number of evacuation nodes
m is the number of relevant transit arcs
capacity_y is the capacity of transit arc y
k_y is the number of population groups transiting by this arc
offset_i_x_y is the date at which population group i_x_y reaches this arc if starting at time 0

The offset is computed adding for each preceding arc (u,v) length(u,v) / speed :

```
l += int(G.edges[u,v]['distance'] / SPEED)
```

Units:
l: minute
distance: 50 m
speed: 50 m / 1 minute

By default, speed=25, hence evacuation speed = 25 * 50 m / minute = 20.8 m/s = speed on the secondary roads

Part IV
Choices for the simulation

3 Roads

The generated road network is composed of 3 types of roads : - secondary roads - primary roads - highways

There is a specific flow for each type of road.

```
secondary_road_flowrate = 70

primary_road_flowrate = 130

highway_flowrate = 200
```

These flow rates have been estimated by the following formula: flow rate = persons/vehicle * speed (m/s) * lane number * vehicles/meter

Whatever the type of road, the number of persons per vehicle is 3. The values of the other parameters are presented in the following table :

Type of road	Colour	Speed (m/s)	Vehicles/m	Lane number
secondary roads	black	20	0.02	1
primary roads	green	25	0.015	2
highways	blue	30	0.0125	3

Type of road	Calculated flow rate (persons/minute)	Used flow rate (persons/minute)
secondary roads	72	70
primary roads	135	130
highways	202	200

Roads are created thanks to two functions. First a road network is generated by the function generate_road_network(limit=800, size=1000).

Then, the function build_roads(G, pos) takes as parameters the return of the first function and creates the different types of road.

In order to have a specific capacity for each arc, the following heuristic is used : capacity (persons/minute)= road_flowrate * (1 + arc_length / land_area_size))

4 Fire propagation

4 program parameters influence the fire propagation : num_iter, intensity, step and firepace

num_iter: Number of wildfire expansion steps (¡int¿ default=100)
intensity: Intensity of the fire (¡float¿ in the interval [0,1], default=.5)
step: Granularity of the fire areas (unit: 50m, ¡int¿, default=10 (ie 500m)
firepace: Time between wildfire expansion (unit: minute, ¡int¿, default=5)

At maximum:
fire speed (m/min) = (step x 50m) / firepace
fire radius (m) = (step * 50m) * num_iter

The higher the intensity is, the closer the wildfire propagation is of these maximum values.

Each wildfire expansion is simulated by the function burn(fire, burnt, visible, wind=(0,0), intensity=.25, step=1, size=1000, foyer=(0,0), direction=(0,0)).

5 Evacuation plan

```
write_evacuation_plan(G, threatened_nodes, safe_zone, num_evacuations,
                  time_factor, filename, tofile=False, writetree=False)
```

The function write_evacuation_plan determines the evacuation tree. It writes a reduced tree (with only relevant arcs) in the file .evac. . If tofile is true, it writes the file .full. If writetree is also true, it adds in the file .full a reduced evacuation tree.

Arguments:
G: networkx graph
threatened_nodes: list of int (list of all the burnt nodes (evacuated nodes and the others))
safe_zone: int (the safe node)
num_evacuations: int (number of evacuations)
time_factor: int (fire pace)
filename: string
tofile: boolean
writetree: boolean

Contents