Usermanual Stride

Episim

2017-2018

Introduction

This manual describes briefly the new features added to the Stride software by the group Episim. First of all, by using the available sample data of cities and communes workplaces, schools, colleges, primary and secondary communities are generated. Then by using some random number generator and on the basis of discrete distribution which are derived from the sample data population is generated. All the cities and communes will have population and according to age and some discrete distribution the population will be assigned to contactpools. The generated population are visualized in GUI. The files of sample data and other parameters which determine the generation of places and population can also be edited by using GUI. The generated population and places can be saved in a file and read and regenerated. The important characteristics of the population are visualized by using charts.

1 Software

1.1 System Requirements

The software is written in **C++** and some of the GUI aspect is written in **QML (Qt Modeling language)** which is a user interface markup language. The following tools are required for the installation of the software:

- g++ or clang compiler

- CMake

- Boost

- Qt5 (for GUI only)

- OpenMP(optional)

- gtest (for testing purpose)

1.2 Installation

The project can be downloaded or cloned from the Github repository. It can be installed by using CMake. The conventional make targets to "build", "install", "test" and "clean" are available. The more details on building the software can be found in "INSTALL.TXT" in the source folder.
By default all the executables are saved in **installed**/**bin** directory inside the project. One can modify by using CMakeLocalConfig.txt file.

1.3 Testing

More tests are added for the added new features. The Google's **gtest** framework is used for the testing. The following commands can be used for testing:
To run all the tests:

- `make test` or

- `./bin/gtester`

To run the tests for the features added by our group only:

- `make episim-test` or

- `./bin/gtester-episim`

`"--gtest_output=xml:<filename>.xml"` flag can be used to generate a xml file with all the details. These can be summarized in html file using *TestSummarizer* class located in */src/main/cpp/popgen-Episim/util* directory.

2 Simulator

The simulator can be started normally using default configuration by using the command.

```
./bin/stride
```

The following flags can be used for other options.

- -p <none|step|end>, –png <none|step|end> Indicates whether or not snapshots should be taken from the map. By default, no snapshots will be taken. If step is chosen, an image will be written to the output folder within a separate folder called 'png'. Each image will be called X.png where X represents the day that was simulated. The end option will only take a snapshot at the end of the simulation. Note that this will automatically introduce a syntheticly generator population since the populations for the regular simulator do not have geographical information.|

- -m <none|step|end>, –mapviewer <none|step|end> Indicates which option should be passed to the mapviewer. By default, the mapviewer will not be used. When using the step option, a map will be showed during each step of the simulation. The end option will only show the map at the end of the simulation. Note that this will automatically introduce a syntheticly generator population since the populations for the regular simulator do not have geographical information.

- **-e** <clean|dump|sim|simgui|geopop>, –exec <clean|dump|sim|simgui |geopop> Execute the function selected.

  clean: cleans the configuration file and writes it to a new file.

  dump: takes the built-in configuration writes it to a file.

  sim: runs the simulator and is the default.

  simgui: runs the simulator with a graphical interface.

  geopop: runs the geospatial synthetic population generator

- **-c** <CONFIGURATION>, –config <CONFIGURATION> Specifies the run configuration parameters. The format may be either -c file=<file> or -c name=<name>. The first is most commonly used and may be shortened to -c <file>. The second format refers to built-in configurations specified by their name.

- **-o** «NAME>=<VALUE», –override «NAME>=<VALUE» (accepted multiple times) Override parameters in the configuration specified with –config.

- **-i**, –installed Look for configuration file specified by the -c file=<file> or -c <file> in the stride install directories

- –**stan** <COUNT> Stochastic Analysis (stan) will run <COUNT> simulations, each with a different seed for the random engine.This option only applies in case of -e sim. It effects overrides to the configuration. See the manual for more info.

- –**version** Displays version information and exits.

- **-h**, –help Displays usage information and exits.

3 Configuration file

The geogrid depends few parameters like total population, age distribution fractions, etc. This information is given by using a configuration file which is in xml format. The data files that are used, the size of contact pools, how the cities are defragmented. These informations are given in that file. An example of this file **geogen_default.xml** can be found in **main**/**resources**/**config**/ directory.
This can be edited by using our GUI as well.

4 Charts

The important characteristics of the generated population can be visualized in Qt Charts. The following command can be used:

```
./bin/Popchart
```

The following characteristics are visualized:

- *Age distribution*: In a barchart the number of people in different age categories are displayed.

- *Population density*: In a pie-chart different population groups are displayed.

- *Household distribution*: In a barchart the number of household with different number of members are displayed.

- *Workplace distribution*: Workplaces are divided as micro, small, medium and large according to the number of staffs in a workplace. In a barchart the number of different types of workplaces are displayed.