**How to run the simulator to obtain the results in the article**

Masaya M. Saito (University of Nagasaki, Siebold)

1. **Structure of the repository**

This repository dist-ibm-covid19 (URL: <https://github.com/mmsaito/dist-ibm-covid19>) provides programs used in article “Effects of inbound attendees of a mass gathering event on the COVID-19 epidemic using individual-based simulations.”

This repository consists of the following three sub-repositories:

* **ibm-saitohm-f90** (URL: <https://github.com/mmsaito/ibm-saitohm-f90/tree/b219ca8d92276c1d5232704c50b60ed57f8af0e5>)

provides the main simulator. Written in Fortran.

* **make-city** (URL: [mmsaito/make-city at dfacdba64f348c536c9bb58f9bb704921dfb1278 (github.com)](https://github.com/mmsaito/make-city/tree/dfacdba64f348c536c9bb58f9bb704921dfb1278))

provides sub programs to make a model city, which is fed to the main simulator. Written in Standard ML.

* **run5-3b-big-take3**　(URL: <https://github.com/mmsaito/run5-3b-big-take3/tree/af6342fb0face4b147d7848712bccdb4fe1f488e>)

provides a template of working directory for a productive run of the simulator and contains configurations files for the simulator.

1. **Procedure to configure the programs**
   1. **Build the simulator**

The simulator is written in Fortran. NVIDIA GPU board and NVIDIA C Compiler (nvc) are required to build the code. Indeed, by the nature of nvc, the simulator may run on a standard Intel/AMD CPU. But we did not fully test where the simulator yields correct results in such a environment.

An executable of the simulator is build by executing make command in the simulator directory:

|  |
| --- |
| cd ibm-saithom-f90  make -f Makefile.pgi |

Then You have an executable **jcl\_sh1.pgi**. To run this executable, you need to pass several items as command-line arguments. So, leave the code at this stage.

* 1. **Build a model city**

You have a model city generator in directory **make-city**. This is written in Standard ML and you need to install Standard ML of New Jersey (<https://www.smlnj.org/>). As the developer is stated, it is advisable to install “legacy version” with version 110.99.xx.

The following steps build a model city:

|  |
| --- |
| cd make-city  sml  (\* commands below are for Standard ML \*)  use ”make-city-with-outer.sml” |

You will have **make-city/scale1.0/model-with-outer-nVis#50000.city**. In order to run the simulator as it is in the later steps, please here make a symbolic link to this file:

|  |
| --- |
| cd scale1.0/  ln -s model-with-outer-nVis#50000.city model-with-outer.city |

The suffix nVis#50000 means the number of visitors is 50,000. To change this number, open make-city-with-outer.sml and edit the following line accodingly:

|  |
| --- |
| val {city, conf0} = makeCityWithVisitor 50000; |

1. **Run the simulator**

We have prepared a template of working directory for running simulators with a specific configuration. The directory includes the following files:

* **run\_test.sh** shell script to run the simulation. This is actually used in our computation to submit the jobs in supercomputer. By reading this file, you may see what arguments are required by the simulator executable **jcl\_sh1.pgi**.
* **inf200.csv** configures initially infected people and each line define when (in minutes), in what town, and what individual set to be infected.
* **extra.cfg** configures infection risks in the main venue and at secondary gathering.

If your computer system does not employ job management system, you can start the simulation by executing run\_test.sh in the working directory, otherwise please consult your system’s manual. After the simulation runs are finished (by default, you will have the results of 32 runs with different random number seeds), your have files with a name **pop\_pgi\_*yymmdd*\_*HHMMSS*\_*nn*.csv** and each of them has a following format:

|  |
| --- |
| t,s,e,i,r,s,e,i,r,s,e,i,r,s,e,i,r,s,e,i,r,  60,575961, 0, 0, 0,177826, 0, 0, 0,139944, 0, 0, 0,316361, 0, 0, 0, 45760, 0, 0, 0, 50060, 0, 0, 0,  ...  1440,575868, 93, 0, 0,177797, 29, 0, 0,139921, 23, 0, 0,316310, 51, 0, 0, 45756, 4, 0, 0, 50060, 0, 0, 0,  1500,575868, 93, 0, 0,177797, 28, 1, 0,139921, 23, 0, 0,316310, 51, 0, 0, 45756, 4, 0, 0, 50060, 0, 0, 0,  1560,575868, 91, 2, 0,177797, 28, 1, 0,139921, 23, 0, 0,316310, 48, 3, 0, 45756, 4, 0, 0, 50060, 0, 0, 0,  1620,575868, 90, 3, 0,177797, 26, 3, 0,139921, 23, 0, 0,316310, 48, 3, 0, 45756, 4, 0, 0, 50060, 0, 0, 0,  1680,575868, 88, 5, 0,177797, 26, 3, 0,139921, 23, 0, 0,316310, 47, 4, 0, 45756, 4, 0, 0, 50060, 0, 0, 0,  1740,575868, 86, 7, 0,177797, 26, 3, 0,139921, 22, 1, 0,316310, 47, 4, 0, 45756, 4, 0, 0, 50060, 0, 0, 0,  1800,575868, 86, 7, 0,177797, 26, 3, 0,139921, 22, 1, 0,316310, 47, 4, 0, 45756, 4, 0, 0, 50060, 0, 0, 0,  1860,575868, 82, 11, 0,177797, 26, 3, 0,139921, 22, 1, 0,316310, 46, 5, 0, 45756, 3, 1, 0, 50060, 0, 0, 0,  1920,575867, 82, 12, 0,177797, 25, 4, 0,139921, 22, 1, 0,316310, 44, 6, 1, 45756, 3, 1, 0, 50060, 0, 0, 0,  1980,575867, 81, 13, 0,177797, 25, 4, 0,139921, 22, 1, 0,316310, 44, 5, 2, 45756, 3, 1, 0, 50060, 0, 0, 0,  2040,575866, 81, 14, 0,177797, 24, 5, 0,139921, 22, 1, 0,316310, 43, 6, 2, 45756, 3, 1, 0, 50060, 0, 0, 0,  2100,575866, 80, 15, 0,177797, 24, 4, 1,139921, 22, 1, 0,316310, 42, 7, 2, 45756, 3, 1, 0, 50060, 0, 0, 0,  2160,575866, 79, 16, 0,177797, 24, 4, 1,139921, 22, 1, 0,316310, 42, 7, 2, 45756, 3, 1, 0, 50060, 0, 0, 0,  2220,575866, 78, 17, 0,177797, 24, 4, 1,139921, 22, 1, 0,316310, 41, 8, 2, 45756, 3, 1, 0, 50060, 0, 0, 0,  2280,575866, 77, 18, 0,177797, 24, 4, 1,139921, 22, 1, 0,316310, 40, 9, 2, 45756, 3, 1, 0, 50060, 0, 0, 0,  2340,575866, 76, 19, 0,177797, 24, 4, 1,139921, 21, 2, 0,316310, 40, 9, 2, 45756, 3, 1, 0, 50060, 0, 0, 0,  ... |

Each line records time (in minutes), the numbers of susceptible, exposed, infectious, and removed agents in town 1, and the similar numbers in the rest of towns. By gathering many simulation runs to evaluate e.g. quantile points, you have epidemic curves show in Fig. 2 of the article.