Running the simulation

Requirements:

- Java 1.8 or newer
- Excel 2010 or newer

Run instructions:

First time run

The simulator generates the following project structured on its first run:

```
parent/ simulator.jar input/ structured.xlsx unstructured.xlsx logs/ output/
```

To generate the excel files used for simulation configuration either execute MAKE*EXCEL.sh (or MAKE*EXCEL.bat when using Windows) or execute the simulator with the `--generate-excel` flag as described below.

Run configuration

- First enter the simulation parameters in the excel files found in the `input` folder.
- Execute the included shell scripts
  - To run a structured simulation execute STRUCTURED.sh (or STRUCTURED.bat when using Windows)
  - To run an unstructured simulation execute UNSTRUCTURED.sh (or UNSTRUCTURED.bat when using Windows)
- It is also possible to drag multiple excel files onto the shell scripts for performing consecutive simulations.

OR

- Run 'java -jar program/simulator.jar' from the parent folder
- To execute in debug mode use the flag `--debug` or `-d`
- To generate a .mol file add `--mode=structured` or `-m=structured`
- To generate a molecular weight distribution add `--mode=unstructured` or `-m=unstructured`
- To generate the Excel file for the given input mode add `--generate-excel` or `-ge`

Input configuration

The simulation parameters can simply be entered in the Excel files. The interaction radius function, which is one of the parameters, uses the following format: It converts a species described by the tuple <I,M,C,P,EN,EC,MC> to a radius (in nm). The mathematical equation that can be entered may contain the following elements:

- `I`: number of half-initiators in molecule
- `M`: number of (non-crosslinker) monomers in molecule
- `C`: number of crosslinkers in molecule
- `P`: number of (pendent) vinyl groups in molecule
- `EN`: number of reactive centers on chain end non-crosslinker molecule
- `EC`: number of reactive centers on chain end crosslinker
- `MC`: number of reactive centers on mid-chain crosslinkers
- `R`: number of reactive centers in molecule
- `w`: total weight of molecule
- `l`: length of a monomer segment
- `n`: total number of monomer segments, equivalent to `I+M+2C-P`
- `m`: total number of crosslinks, equivalent to `C-P`. Use `R-1` if you only want intermolecular crosslinks
- `pi` and `π`: pi
- `(` and `)`: left and right parenthesis
- both integers and decimal numbers with the `.` separator

and the following operators:

- `-` for subtraction

- `+` for addition
- `*` for multiplication
- `/` for division
- `^` for powers
- `√` and `sqrt()`: for square roots

Examples:

- `l√(n/6) (√(1+m/6)+(4m/(3p)))^-0.25` by [1]
- `0.0144w^0.561` by [2]

Output configuration

Custom output functions are used to create a data point for a set of polymers, which are again represented by the aforementioned tuples. These functions use the same elements as those discussed in the previous section. Since it is not feasible to create a data point per polymer in the simulation, we offer additional modifiers to aggregate the results:

- `SUM()`: applies the function to every polymer in the simulation and adds the results together
- `AVG()`: applies the function to every polymer in the simulation and calculates the average value
- `WAV()`: applies the function to every polymer in the simulation and calculates the weight average
- `ZAV()`: applies the function to every polymer in the simulation and calculates the z-average

One of these modifiers has to be used. Additionally, these modifiers can be combined with any number of the following ones:

- `BIN()`: calculate a value per weight category, rather than one value for the whole set of molecules
- `INC()`: include starting molecules (half-initiators, monomers and crosslinkers) in the data as well
- `EXC()`: exclude the biggest molecule in the data

Several alias have been created for easier entering of output functions:

- `MN` : number average mass, equivalent to `AVG(w)`
- `MW` : weight average mass, equivalent to `WAV(w)`
- `MZ` : weight average mass, equivalent to `ZAV(w)`
- `MND`: monomer number distribution, equivalent to `BIN(SUM(1))`
- `MWD`: monomer weight distribution, equivalent to `BIN(SUM(w))`
- `MZD`: monomer z-distribution, equivalent to `BIN(SUM(w^2))`

and additionally

- `SIZE`: equivalent to twice the interaction radius
- `PDI`: equivalent to `MW/MN = WAV(w)/AVG(w)`, which cannot be entered via equation

These can be combined with the `EXC()` and `INC()` modifiers. For example, if we want to calculate the z-average size per weight category, including the initial molecules, but excluding the largest molecule, we would use the function `EXC(INC(BIN(ZAV(SIZE))))`.

Compiling the simulation

Requirements:

- Java Development Kit 1.8+
- Maven 3.6.1+
- Add javac to the Windows PATH variable
- Add mvn to the Windows PATH variable

Instructions:

- Run the command `mvn package` in the root folder of the project to create the .jar file.
- Rename the generated jar from `Simulator-1.0-stand-alone.jar` to `simulator.jar` and place in folder with shell scripts, if you want to be able to use those.

Testing the simulation

- Java assertions can be enabled by using the `-ea` flag when running the simulation using the `java` command. This can be done in IntelliJ via `Run -> Edit Configurations -> VM options`.
- Tests can be executed using the `mvn verify` command. This only executes validation tests, not speed tests and logging tests.

References

[1]: H. Tobita, "Dimensions of Cross-Linked Polymers Formed in Living Vinyl/Divinyl Copolymerization", *Macromolecules*, vol. 27, no. 19, pp. 5413-5420, 1994.

[2]: J.A. Pomposo, I. Perez-Baena, F. Lo Verso, A.J. Moreno, A. Arbe, and J. Colmenero, "How far are single-chain polymer nanoparticles in solution from the globular state?", *ACS Macro Letters*, vol. 3, no. 8, pp. 767-772, 2014.