ParCell Documentation

Table of Contents

List of Figures

List of Tables

Introduction

The interface of the ParCell is based on paper [1, 2].

Installation procedure

1) Create a folder and copy all the files of ParCell in that folder. Here, a folder called 'GUI' is created. Folder name 'GUI' is replaceable by user defined folder name.
2) Copy the GUI folder into Linux-based system.
3) Python 3, numpy and matplotlib needs to be installed from terminal if not available.

Run exe file

From terminal go to the folder '/GUI/dist'. Run the following command (recommended).

./gui

Note: Go to folder 'GUI/dist' and double click of application file will also run the interface.

ParCell Interface

The interface will open with two windows. First, provide the path of folder GUI in 'set path' window. Then press 'Set' button. This will set the folder GUI as current directory. Then close the window by pressing 'close' or 'x' at corner.

The main window of ParCell contains 5 buttons as shown in Figure 1.

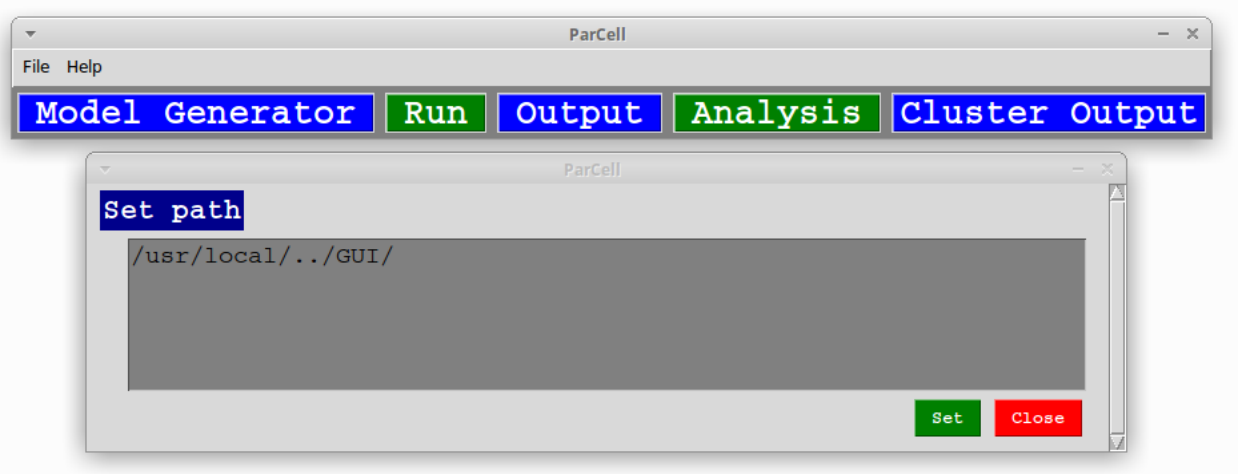

Figure 1: ParCell Interface

Model Generator

Figure 2: ParCell Editor

This editor is for model rules input. ‘Compile Model’ button will compile the rules. ‘Save’ button will save the model file in user provided name location as ‘.pcl’ file. Multiple windows can be opened at the same time. Figure 2 shows the ParCell editor.

Shortcut keys for Model generator text editor

Copy: control + c
Paste: control + v
Undo: control + z
Select section: hold mouse left button, drag over the selected area.

Load an existing model

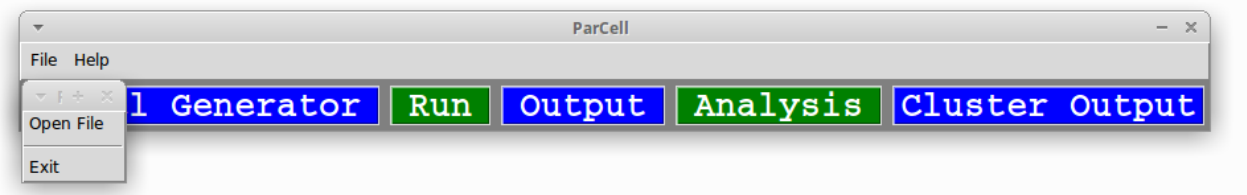

Figure 3: File tab of ParCell from Menu Bar

Go to File at top left corner in menu bar and press ‘Open File’. Go to the model file and select, open the model file in the Parcel editor. ‘Exit’ will quit the software. Figure 3 shows the File tab.

Run

Once compile the rules in Parcel editor, press ‘Run’ in the main window to run the simulation. The software will remain freeze and the ‘Run’ button will remain white while simulation is running. It will turn green again once the simulation is over

Output File

Dynamic output can be seen in terminal by going to folder ‘/GUI/BIBM/Output/’ and use the following command.

vi logfile.pcl

It will open the output file. The output file in terminal is closed by following keys in keyboard.

shift + z + z

Once the simulation is over the output can be seen by pressing ‘Output’ button in main window. It will open ‘logfile.pcl’ in gedit (recommended). Figure 4 shows the Output window of ParCell.

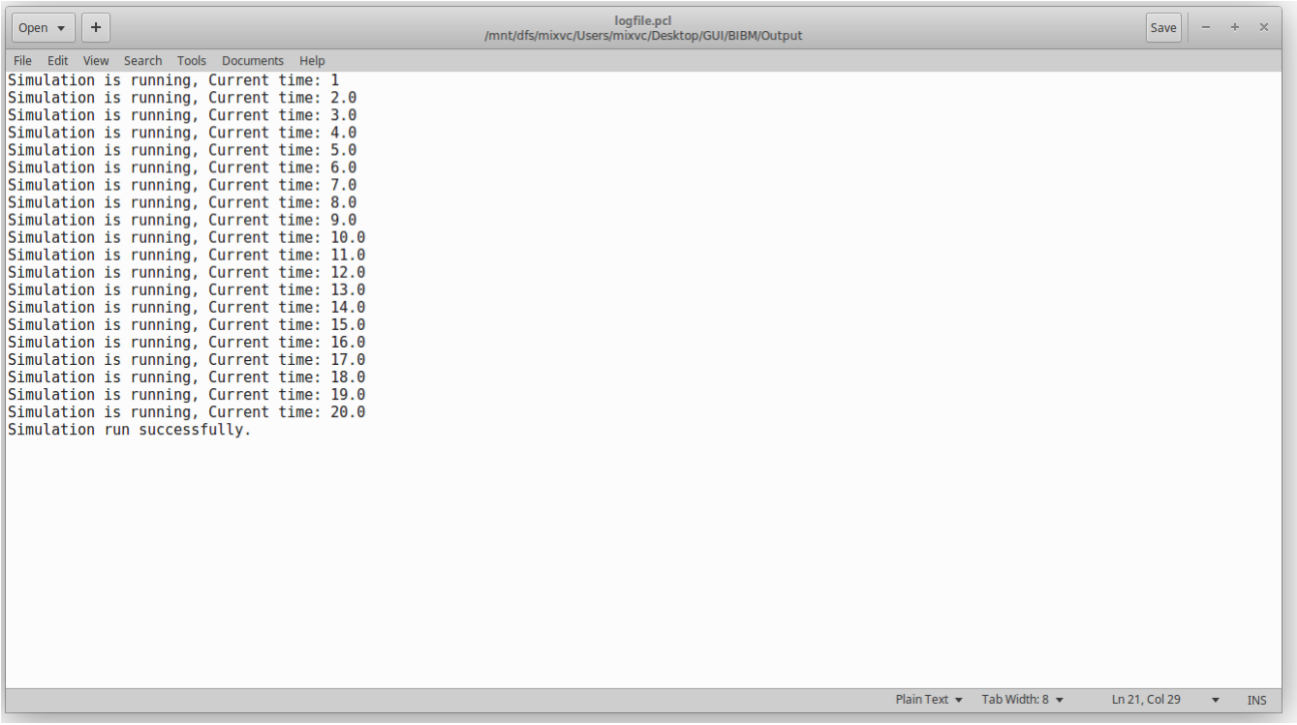

Figure 4: Output Window of ParCell

Note: Outputs of the model are available in ‘GUI/BIBM/model’.

Analysis

Pressing ‘Analysis’ button in the main window will open analysis tab. ParCell analysis tab is built on library ‘matplotlib’ of python 3. Figure 5 shows the analysis editor.

Figure 5: ParCell Analysis editor

Cluster Output

Pressing Cluster Output button will create a zip file in ‘GUI/BIBM/Output’ folder. Upload the file in cluster and run the following command in scheduler.

python Main.py

The following libraries may be required:

1) Numpy

The Output of the model file can be analyzed using the analysis tab by simply replacing the Output folder with Output from cluster.

Model Rules in ParCell Editor

Pressing ‘Model Generator’ button in Main window will open ParCell editor. ParCell editor is used to writing the model. Figure 6 shows a schematic diagram of model rules. The model rules require four basic blocks as mandatory inputs: parameter, species, initial condition, and reaction rules. Dependent reaction rules block is an extension of reaction rules block. Environmental species block is for defining environmental molecules and environmental reaction rules block is for specific rules of environmental molecule. Other blocks are cellular birth, cellular death, and simulation end rules blocks. They are connecting cell function to subcellular network dynamics. All the blocks are to be kept within ‘begin model’ and ‘end model’ command. ‘simulate()’ function

is called outside the ‘end model’. ‘#’ is used for commenting. Figure 7 shows a template of all the blocks.

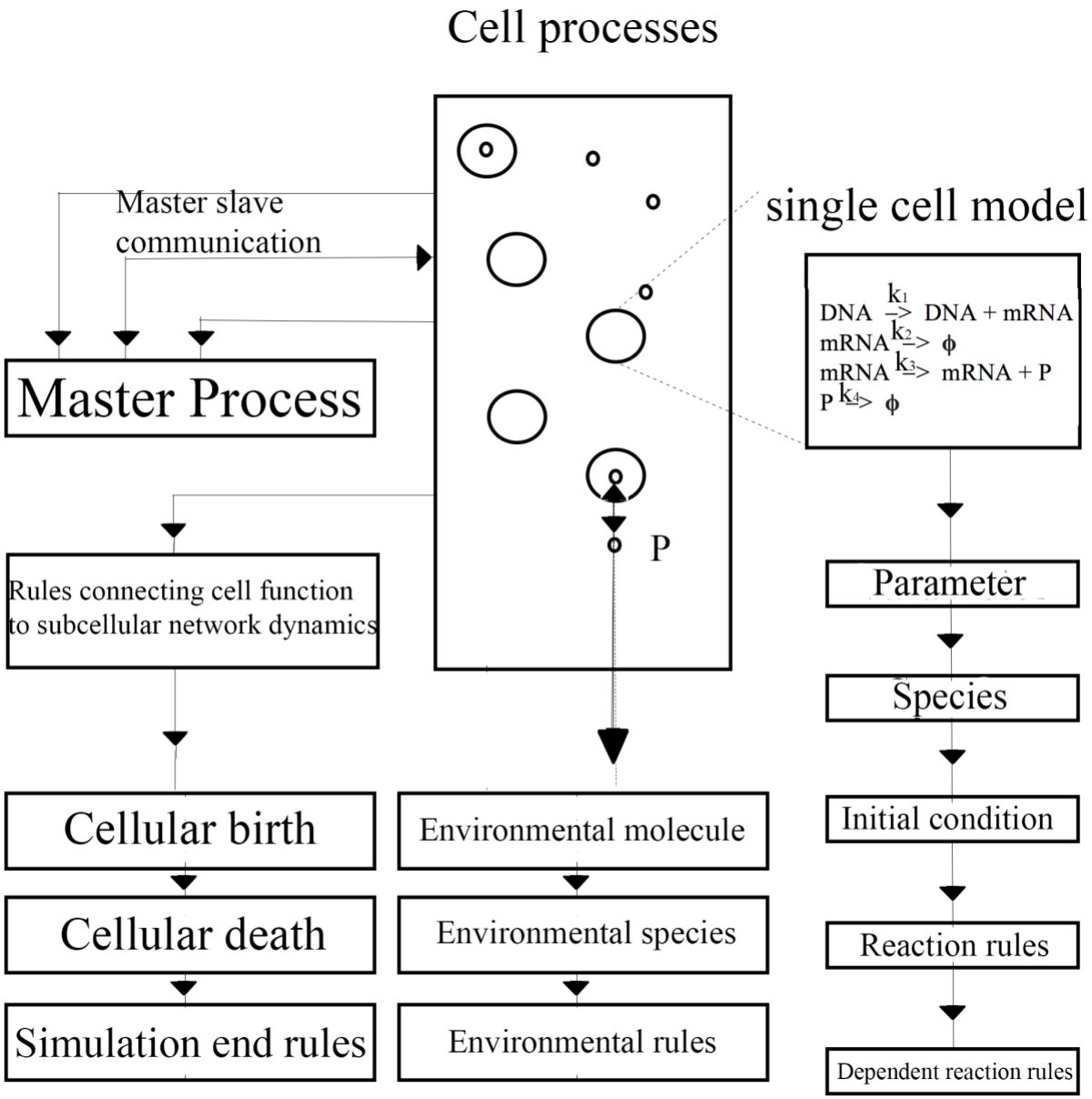

Figure 6: Schematic diagram of ParCell model generator

```
begin model

# Parameter block
begin parameter
end parameter

# Species block
begin species
end species

# Initial condition block
begin initial condition
end initial condition

# Reaction rules block
begin reaction rules
end reaction rules

# Dependent reaction rules block
begin dependent reaction rules
end dependent reaction rules

# Environmental species block
begin environmental species
end environmental species

# Environmental rules block
begin environmental rules
end environmental rules

# Cellular birth block
begin cellular birth
end cellular birth

# Cellular death block
begin cellular death
end cellular death

#Simulation end rules block
begin simulation end rules
end simulation end rules

end model

simulate()
```

Figure 7: Template of model rules in ParCell Editor

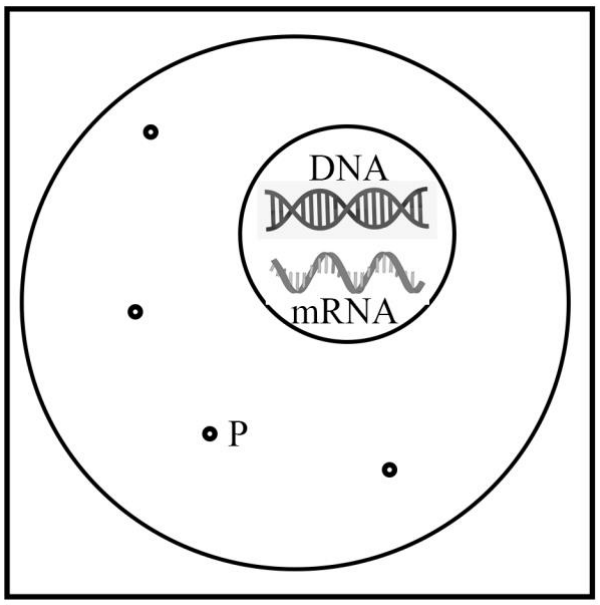

Figure 8: Transcription and translation of DNA and mRNA into protein P

Figure 8 shows a simple model of transcription of DNA into mRNA and translation of mRNA into protein P. This system can be defined completely by the basic blocks of ParCell as described in following sections.

Parameter Block

```
begin model

# Parameter block
begin parameter

b   20          # Burst factor

k1  0.01        # Transcription of DNA into RNA
k2  0.00577     # Degradation of mRNA
k3  k2*b        # Translation of RNA into protein P
k4  0.0001925   # production of mRNA

end parameter
```

Figure 9: Parameter block

Figure 9 shows a sample copy of parameter block. All the parameters are listed within 'begin parameter' and 'end parameter'. First column stored names, and second column stored the values. Table 1 shows the available math operator in ParCell editor.

Table 1: Math operators available in ParCell Editor

Name	Math operator
Multiply	*
Division	/
addition	+
substation	-
exponent	e
bracket	()

Note: Don't put spaces when writing the name of the parameters in first column.

Species Block

```
# Species block
begin species

DNA
mRNA
P

end species
```

Figure 10: Species block

Figure 10 shows a sample copy of species block. The block starts with 'begin species' and ends with 'end species'. Name of all the species in all compartments of cell and tissue are listed here. Figure 11 shows the exchange of P molecule between cell and its environment. If same molecule exists in different compartment then first bracket and short name for compartment can be used.

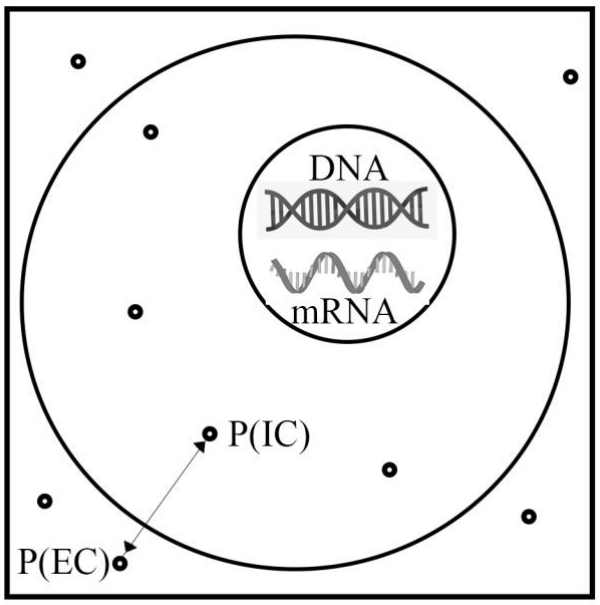

Figure 11: Diffusion of protein P from cell to its environment. The intracellular and extracellular P are denoted as P(IC) and P(EC) respectively.

Figure 12 shows species block in compartmentalized form where P(IC) is the intracellular P and P(EC) is the extracellular P.

```
# Species block
begin species

DNA
mRNA
P(IC)
P(EC)

end species
```

Figure 12: Compartmentalized species block

Note: Don't put spaces when writing the name of the parameter.

Initial condition block

The block starts with 'begin initial condition' and ends with 'end initial condition'. All the initial values of the species are listed here. First column stored names, and second column stored the initial values. Example copy of initial condition block is shown in Figure 13.

```
#3. Initial condition block
begin initial condition

DNA  1

end initial condition
```

Figure 13: Initial condition block

Reaction rules block

The block starts with 'begin reaction rules' and ends with 'end reaction rules'. It has four columns and separated by '$', '$$' and '$$$'. Format is [Reaction rules $ rate constant $$ distribution $$ dependent reaction tag].

Column 1: Reaction rules

The reactants are in left side of (->), and the products are in the right side. For multiple species ('+') sign is used. Some of the case scenarios are listed in Table 2.

Table 2: Case scenarios of different reaction rules

Case	Rules	Comments
Case 1	**A -> B**	reactants and products are separated by '->'
Case 2	**A + B -> C**	two reactants are separated by '+' sign
Case 3	**A -> B + C**	two products are separated by '+' sign
Case 4	**A ->**	Decay of reactant A
Case 5	**-> A**	if produced from DNA or external supply
Case 6	**A + A -> A_2**	instead of 2A write A + A
Case 7	**A -> B** **B -> A**	write elementary form for bidirectional reaction
Case 8	**A + A -> A_2**	Rate constant = 0.5*ka*$[A]_2$ format is considered. Here, ka is rate constant.

Column 2: Rate constant

The name of the rate constant for each reaction is listed in column 2 from 'parameter block'. Column 1 and 2 are separated by '$' command.

Column 3: Distribution

This function defines the distribution type of a parameter among all cells. This will incorporate cellular heterogeneity by fluctuation rate constant parameter. Column 2 and 3 are separated by '$$' command. The available distribution functions are shown in Table 3.

Table 3: Distribution functions

Distribution type	function	comment
constant	**(constant)**	The parameter will be constant across all cells
uniform	**(uniform, low value, high value)**	The parameter will be distributed uniformly across all cells between high and low value. Example: (uniform, 10, 50))
normal	**(normal, mean value, standard deviation)**	The parameter will be distributed normally across all cells with provided mean and standard deviation value. Example: (normal, 30, 0.1)
lognomal	**(lognormal, mean value, standard deviation)**	The parameter will be distributed lognormally across all cells with provided mean and standard deviation value. Example: (lognormal, 1e-3, 0.1)

Column 4: Reaction tag

This column is used to tag a reaction to modify the propensity of reaction rules dynamically. The modified rules are defined in 'dependent reaction rules' block. The available tags are DRR1 to DRR8. Multiple reaction can be tagged with a single tag (e.g.: two reactions with DRR1). Details are available in 'dependent reaction rules block'. Column 3 and 4 are separated by '$$$' command.

Note: Use '$$$' command only to tag reactions. It is not mandatory.

Some examples of reaction rules block are as follows.
Example 1:
In Figure 14, first column is reaction rules, second column rate constant parameter, and third column distribution type. (constant) function will keep the rate constant parameter same for all cells.

```
# Reaction rules block
begin reaction rules

# 1. Transcription of DNA into mRNA
DNA -> DNA + mRNA          $ k1       $$ (constant)

# 2. Degradation of mRNA
mRNA   ->                  $ k2       $$ (constant)

# 3. Translation mRNA into protein P
mRNA   ->   mRNA  +  P     $ k3       $$ (constant)


# 4. Degradation of protein P
P    ->                    $ k4       $$ (constant)
```

Figure 14: Reaction rules block

Example 2:
Figure 15 shows the reaction rules block at different distribution of rate constant parameter.

```
# Reaction rules block
begin reaction rules

# 1. Transcription of DNA into mRNA
DNA -> DNA + mRNA          $ k1       $$ (constant)

# 2. Degradation of mRNA
mRNA   ->                  $ k2       $$ (uniform, 0.0052, 0.0062)

# 3. Translation mRNA into protein P
mRNA   ->   mRNA  +  P     $ k3       $$ (constant)


# 4. Degradation of protein P
P    ->                    $ k4       $$ (normal, 0.0001925, 0.1)
```

Figure 15: Different distribution of rate constant parameter across all cells

Simulate Function

To run the simulation, at the end of the model (after 'end model' command) following function is called with system parameters.

```
simulate()
```

Parameters name and default values of the function are listed in Table 4.

Table 4: Default values of simulate function

Description	Parameter Name	Default Value
Total Number of Cells	N	2
Total Simulation Time	T	100
Number of Cores	cores	4
Time interval between Master Slave communication	grain	1
Data save interval	grainWrite	1
Mean division time	meanDivisionTime	None
Mean death time	meanCellDeath	None
Fluctuation of total number of cells	KValue	2

The parameters of the simulate function can be changed as follows:

```
simulate(N = 20, T = 500, cores = 8, grain = 0.1, grainWrite = 0.1)
```

'N' denotes the total number of cells and 'T' denotes the total simulation time. The number of parallel processes for simulation can be set by parameter 'cores'. Parameter 'grain' set the interval between Master Slave communication and 'grainWrite' set the interval at which simulated data will be saved.

Parameter 'meanDivisionTime' will introduce cellular birth death keeping the cell density constant. The division time for all cells will be set from an exponential distribution based on the mean division time (Weber 2013).

```
simulate(N = 20, T = 500, cores = 8, meanDivisionTime = 45)
```

Parameter 'meanCellDeath' will introduce cellular death. The death time for all cells will be set from an exponential distribution based on the mean cell death time.

```
simulate(N = 20, T = 500, cores = 8, meanCellDeath = 30)
```

Parameter 'KValue' defines the maximum number of cells for a dynamic population. Maximum number of cells for the system is Total number of cells N times KValue. If the initial cell population N is 20 and KValue is 2 then the maximum number of cells for the system will be 40.

```
simulate(N = 20, T = 500, cores = 8, KValue = 5)
```

In the above example, maximum number of cells for the system is N × KValue which is 100.

$$Maximum\ number\ of\ cells = \text{N} \times \text{KVlaue} = 100$$

Dependent reaction rules block

This block provides additional rules in the calculation of tagged reactions propensity in 'reaction rules' block. It starts with 'begin dependent reaction rules' and ends with 'end dependent reaction rules'. The software allows 8 set of modified rules/ functions. Use '**[tag name]*=**' command to access the function and use modified propensity by multiplying its new rules/functions.. Available tag names are: DRR1, DRR2, DRR3, DRR4, DRR5, DRR6, DRR7, DRR8. If multiple reactions follow same modified rule, they can be tagged with same tag name. Figure 16 shows an example of dependent reaction block. In the example, propensity of reaction 3 and reaction 4 will be multiplied by 2.

```
# Reaction rules block
begin reaction rules

# 1. Transcription of DNA into mRNA
DNA -> DNA + mRNA          $ k1   $$ (constant)

# 2. Degradation of mRNA
mRNA   ->                  $ k2   $$ (constant)

# 3. Translation mRNA into protein P
mRNA   ->   mRNA  +  P     $ k3   $$ (constant)   $$$ DRR1


# 4. Degradation of protein P
P    ->                    $ k4   $$ (constant)   $$$ DRR1


end reaction rules

#Dependent reaction block
begin dependent reaction rules

[DRR1] *= 2

end dependent reaction rules
```

Figure 16: Dependent reaction rules block

Figure 17 shows the use of dependent reaction rules block for partial distribution of one parameter. As shown in figure 7 translation of mRNA to protein occur with burst and also depend on the rate constant parameter of degradation of mRNA. As shown in Figure 13 we can first distribute the burst factor [3] among all cells first, and then the constant part of the parameter can be included in the dependent reaction rules block. This way partial of the rate constant parameter will be distributed among cells.

```
# Parameter block
begin parameter

b     20                # DNA Transcription

k1    0.01              # Transcription of DNA into RNA
k2    0.00577           # Degradation of mRNA
k3    b                 # Translation of RNA into protein P
k4    0.0001925         # production of mRNA

end parameter

# Reaction rules block
begin reaction rules

# 1. Transcription of DNA into mRNA
DNA -> DNA + mRNA         $ k1   $$ (constant)

# 2. Degradation of mRNA
mRNA   ->                 $ k2   $$ (constant)

# 3. Translation mRNA into protein P
mRNA   ->   mRNA  +  P    $ k3   $$ (normal, 20, 1)   $$$ DRR1


# 4. Degradation of protein P
P    ->                   $ k4   $$ (constant)


end reaction rules

#Dependent reaction block
begin dependent reaction rules

[DRR1] *= 0.00577

end dependent reaction rules
```

Figure 17: Dependent reaction rules block with partial distribution of one parameter

Figure 18 shows the rules for dependency of rate constant parameter on dynamic evolution of molecule. Parameter 'moleculeA[species name]' is used in the function to define the dependency. Figure 18 shows the function where the degradation of protein P depends on the molecular number of mRNA over time (mock model).

```
# Parameter block
begin parameter

b     20                # DNA Transcription

k1    0.01              # Transcription of DNA into RNA
k2    0.00577           # Degradation of mRNA
k3    b                 # Translation of RNA into protein P
k4    1                 # production of mRNA

end parameter

# Reaction rules block
begin reaction rules

# 1. Transcription of DNA into mRNA
DNA -> DNA + mRNA         $ k1   $$ (constant)

# 2. Degradation of mRNA
mRNA   ->                 $ k2   $$ (constant)

# 3. Translation mRNA into protein P
mRNA   ->   mRNA  +  P    $ k3   $$ (constant)


# 4. Degradation of protein P
P    ->                   $ k4   $$ (constant)   $$$ DRR1


end reaction rules

#Dependent reaction block
begin dependent reaction rules

[DRR1] *= (100 + moleculeA[mRNA]*2)/(100 + moleculeA[mRNA])

end dependent reaction rules
```

Figure 18: Dependency of rate constant parameter on dynamic evolution of molecule

Note: When using multiple tags, maintain order from '**[DRR1]*=**' to '**[DRR8]*=**'
Note: Don't press 'enter' or use line break for the equation.

Environmental species block

In ‘environmental species’ block, the extracellular molecules which all cell share is listed here. It starts with ‘begin environmental species’ and ends with ‘end environmental species’. In ParCell, homogeneous mixture of environmental molecules is considered. At each time interval Master process will synchronize the environmental molecules listed in this block. There is a tradeoff between master-slave communication time (‘grain’) and accuracy in the dynamics of environmental molecules. Smaller ‘grain’ will yield better accuracy but increase computational expenses. The value of ‘grain’ needs to be adjusted if evolution of environmental molecules is rapid. Figure 19 shows the environmental species block. At each time interval master process will evaluate net change between all cells and its environment and pass one net value for environment species to all cells.

```
# Reaction rules block
begin reaction rules

# 1. Transcription of DNA into mRNA
DNA -> DNA + mRNA            $ k1       $$ (constant)

# 2. Degradation of mRNA
mRNA ->                      $ k2       $$ (constant)

# 3. Translation mRNA into intra-cellular protein P
mRNA -> mRNA + P(IC)         $ k3       $$ (constant)

# 4. Degradation of intra-cellular protein P
P(IC) ->                     $ k4       $$ (constant)

# 5. Diffusion of P from intra-cellular compartment to extracellular
compartment
P(IC) -> P(EC)               $ d1       $$ (constant)

# 7. Diffusion of P from extra-cellular compartment to intra-cellular
compartment
P(EC) -> P(IC)               $ d2       $$ (constant)

# 8. Degradation of extra-cellular protein P
P(EC) ->                     $ k4       $$ (constant)

end reaction rules

# Environmental species block
begin environmental species

P(EC)

end environmental species
```

Figure 19: Environmental species block

```
# Reaction rules block
begin reaction rules

# 1. Transcription of DNA into mRNA
DNA -> DNA + mRNA                    $ k1    $$ (constant)

# 2. Degradation of mRNA
mRNA ->                              $ k2    $$ (constant)

# 3. Translation mRNA into intra-cellular protein P1 and P2
mRNA -> mRNA + P1(IC) + P2(IC)       $ k3    $$ (constant)

# 4. Degradation of intra-cellular protein P1
P1(IC) ->                            $ k4    $$ (constant)

# 5. Degradation of intra-cellular protein P2
P2(IC) ->                            $ k4    $$ (constant)

# 6. Diffusion of P1 from intra-cellular comaprtment to extracellular
compartment
P1(IC) -> P1(EC)                     $ d1    $$ (constant)

# 7. Diffusion of P1 from extra-cellular comaprtment to intra-cellular
compartment
P1(EC) -> P1(IC)                     $ d2    $$ (constant)

# 8. Diffusion of P2 from intra-cellular comaprtment to extracellular
compartment
P2(IC) -> P2(EC)                     $ d3    $$ (constant)

# 9. Diffusion of P2 from extra-cellular comaprtment to intra-cellular
compartment
P2(EC) -> P2(IC)                     $ d4    $$ (constant)

# 10. Degradation of extra-cellular protein P1
P1(EC) ->                            $ k4    $$ (constant)

# 11. Degradation of extra-cellular protein P2
P2(EC) ->                            $ k4    $$ (constant)

end reaction rules

# Environmental species block
begin environmental species

P1(EC)
P2(EC)

end environmental species
```

Figure 20: Environmental species block for multiple environmental molecules

Figure 20 shows the environmental rules block when multiple environmental molecules are present. Both of the environmental molecules will be synchronized at each time interval. Any number of environmental molecule can be listed here.

Environmental rules block

This block is an extension of environmental species block. It defines the additional function for environmental molecule or fixed the environmental molecule at constant value (constant external supply) as shown in Figure 21. It starts with 'begin environmental rules' and ends with 'end environmental rules'. Here the parameter 'updateExt[species name]' is used. Additionally, 'moleculeA[species name]' parameter can also be used in the function. In Figure 21, molecular concentration of environmental species P2(EC) will remain constant at 1000 molecules throughout the simulation.

```
# Reaction rules block
begin reaction rules

# 1. Transcription of DNA into mRNA
DNA -> DNA + mRNA                  $ k1    $$ (constant)

# 2. Degradation of mRNA
mRNA ->                            $ k2    $$ (constant)

# 3. Translation mRNA into intra-cellular protein P1 and P2
mRNA -> mRNA + P1(IC) + P2(IC)     $ k3    $$ (constant)

# 4. Degradation of intra-cellular protein P1
P1(IC) ->                          $ k4    $$ (constant)

# 5. Degradation of intra-cellular protein P2
P2(IC) ->                          $ k4    $$ (constant)

# 6. Diffusion of P1 from intra-cellular comaprtment to extracellular
compartment
P1(IC) -> P1(EC)                   $ d1    $$ (constant)

# 7. Diffusion of P1 from extra-cellular comaprtment to intra-cellular
compartment
P1(EC) -> P1(IC)                   $ d2    $$ (constant)

# 8. Diffusion of P2 from intra-cellular comaprtment to extracellular
compartment
P2(IC) -> P2(EC)                   $ d3    $$ (constant)

# 9. Diffusion of P2 from extra-cellular comaprtment to intra-cellular
compartment
P2(EC) -> P2(IC)                   $ d4    $$ (constant)

end reaction rules

# Environmental species block
begin environmental species

P1(EC)
P2(EC)

end environmental species

# Environmental Reactions block
begin environmental rules

updateExt[P2(EC)]  = 1000

end environmental rules
```

Figure 21: Environmental rules block keeping one environmental species constant

Cellular birth block

It starts with ‘begin cellular birth’ and ends with ‘end cellular birth’. The input is one line of condition for cellular birth. It can depend on time (T), number of cells (no_cell) or any evolution of molecules (moleculeA[species name]).

Figure 22 shows that the cell will divide if the number of molecules of protein P is equal or exceeds 300 molecules.

```
# Reaction rules block
begin reaction rules

# 1. Transcription of DNA into mRNA
DNA -> DNA + mRNA          $ k1   $$ (constant)

# 2. Degradation of mRNA
mRNA ->                    $ k2   $$ (constant)

# 3. Translation mRNA into protein P
mRNA -> mRNA + P           $ k3   $$ (constant)

# 4. Degradation of protein P
P ->                       $ k4   $$ (constant)


end reaction rules

# Cellular birth block
begin cellular birth

moleculeA[P] >= 300

end cellular birth
```

Figure 22: Cellular birth block

Figure 23 shows cellular birth with multiple conditions. Any number of condition can be added with 'and' (both conditions needs to be satisfied for cell division to occur), 'or' (if one of the condition satisfied then cellular birth will occur) command. In Figure 23, cell division will occur if simulation time exceeds 200 and number of molecules of protein P in a cell is equal or exceed 300.

```
# Reaction rules block
begin reaction rules

# 1. Transcription of DNA into mRNA
DNA -> DNA + mRNA          $ k1   $$ (constant)

# 2. Degradation of mRNA
mRNA ->                    $ k2   $$ (constant)

# 3. Translation mRNA into protein P
mRNA -> mRNA + P           $ k3   $$ (constant)

# 4. Degradation of protein P
P ->                       $ k4   $$ (constant)


end reaction rules

# Cellular birth block
begin cellular birth

moleculeA[P] >= 300 and T > 100

end cellular birth
```

Figure 23: Cellular birth block with multiple conditions

Cellular death block

This block is similar to cellular birth block. Except instead of division cell death will occur. It starts with 'begin cellular death' and ends with 'end cellular death'. The input is one line of condition for cellular birth. It can depend on time (T), number of cells (no_cell) or any evolution of molecules (moleculeA[species name]). Figure 24 shows an example of cellular death block.

```
# Reaction rules block
begin reaction rules

# 1. Transcription of DNA into mRNA
DNA -> DNA + mRNA          $ k1   $$ (constant)

# 2. Degradation of mRNA
mRNA ->                    $ k2   $$ (constant)

# 3. Translation mRNA into protein P
mRNA -> mRNA + P           $ k3   $$ (constant)

# 4. Degradation of protein P
P ->                       $ k4   $$ (constant)


end reaction rules

# Cellular death block
begin cellular death

moleculeA[P] >= 240 and no_cell >= 200

end cellular death
```

Figure 24: Cellular death block

Simulation end rules block

This block defines additional condition for simulation to end based on number of cells. It starts with 'begin simulation end rules' and ends with 'end simulation end rules'. It is one line of rule defining the condition of simulation end. Parameters are number of cells 'N' and local time 't'. Signs are == (equal), >= (greater than or equal), <= (less than or equal), > (greater than), < (less than).

Figure 25 shows that if number of cells equal to 200 then the simulation will end.

```
#Simulation end rules block
begin simulation end rules

N == 200

end simulation end rules
```

Figure 25: Additional rules to end simulation

Analysis

In the main window, pressing 'Analysis' button will open Analysis windows. Figure 26 shows the ParCell Analysis editor. It has two functions. One is plotter() function to show the trajectory of change of number of molecules over time of all individual cells. Another one is plotterA() function to plot the average response of change of number of molecules over time.

Figure 26: ParCell Analysis editor

The plotter() function has two arrays, first array defining the molecule array of the species and second array defining user defined name of the species. In following example plotter() function will plot number of P molecule over time for all cells.

```
plotter([moleculeA['P']],['Protein P'])
```

Multiple molecules can be plotted in the same plot. Following is an example of plotting number of molecules of DNA and mRNA over time for all cells in a single plot.

```
plotter([moleculeA['DNA'], moleculeA['mRNA']],['DNA', 'mRNA'])
```

The plotterA() function has two arrays, first array defining the molecule array of the species and second array defining user defined name of the species. In the following example plotterA() function will plot the average number of P molecule over time.

```
plotterA([moleculeA['P']],['Protein P'])
```

Multiple molecules can be plotted in the same plot. Following is an example of plotting average number of molecules of DNA and mRNA over time in a single plot.

```
plotterA([moleculeA['DNA'], moleculeA['mRNA']],['DNA', 'mRNA'])
```

To plot, compile the model first (if the model is not compiled), then open analysis editor, write the function, compile it and finally Run it to see the plots.
Cluster Output can directly paste into the Output folder. Then open the model file and compile it. Open plot editor and call the function there.

Note: Use string (‘’) when writing the name of molecule inside moleculeA[] (e.g.: moleculeA[‘P’])

List of errors and warning

Table 5 shows the list of errors and possible reason behind the error.

Table 5: List of Errors and Warnings

Error/Warning message	**Possible Reason**
Error in calling simulate() function. Please use exact name and notation.	Check the name of the parameters. It needs to be exactly as listed in table 4
Error in Parameter block	Check parameter block
Error in Species block	Check species block
Error in Initial condition block	Check initial condition block
Error in Reaction rules block	Check reaction rules block
Error in environmental species block	Check environmental species block
Error in environmental rules block	Check environmental rules block
Error in dependent reaction rules block	Check dependent reaction rules block
Error in cellular birth block	Check cellular birth block
Error in cellular death block	Check cellular death block
Error in simulation end rules block	Check simulation end rules block
Error in defining the cellular birth block, please use proper notation and equation	Condition for cellular birth in cellular birth block is incorrect or not supported
Error in defining the cellular death block, please use proper notation and equation.	Condition for cellular death in cellular death block is incorrect or not supported
Error in defining the environmental rules block, please use proper notation and equation.	Condition for environmental rules in environmental rules block is incorrect or not supported
Error in defining the dependent reaction rules block, please use proper notation and equation.	Condition for dependent reaction rules in dependent reaction rules block is incorrect or not supported
Warning All cells are Inactive. Either reactant is over or cell death occur. Simulation ending.	No active cell due to cell death or the reactant for all cells is over
Warning: Reactant is over for cell id	Reactant for one cell is over

Example Models

Example 1: Transcription of DNA and translation of mRNA

```
begin model

# Parameter block
begin parameter

b     20                # Burst factor

k1    0.01              # Transcription of DNA into RNA
k2    0.00577           # Degradation of mRNA
k3    k2*b              # Translation of RNA into protein P
k4    0.0001925         # production of mRNA

end parameter

# Species block
begin species

DNA
mRNA
P

end species

# Initial condition block
begin initial condition

DNA  1

end initial condition

# Reaction rules block
begin reaction rules

# 1. Transcription of DNA into mRNA
DNA -> DNA + mRNA           $ k1    $$ (constant)

# 2. Degradation of mRNA
mRNA ->                     $ k2       $$ (constant)

# 3. Translation mRNA into protein P
mRNA -> mRNA + P            $ k3       $$ (constant)

# 4. Degradation of protein P
P ->                        $ k4       $$ (constant)

end reaction rules

end model

simulate(N = 20, T = 3000, cores = 4, grain = 1, grainWrite = 1)
```

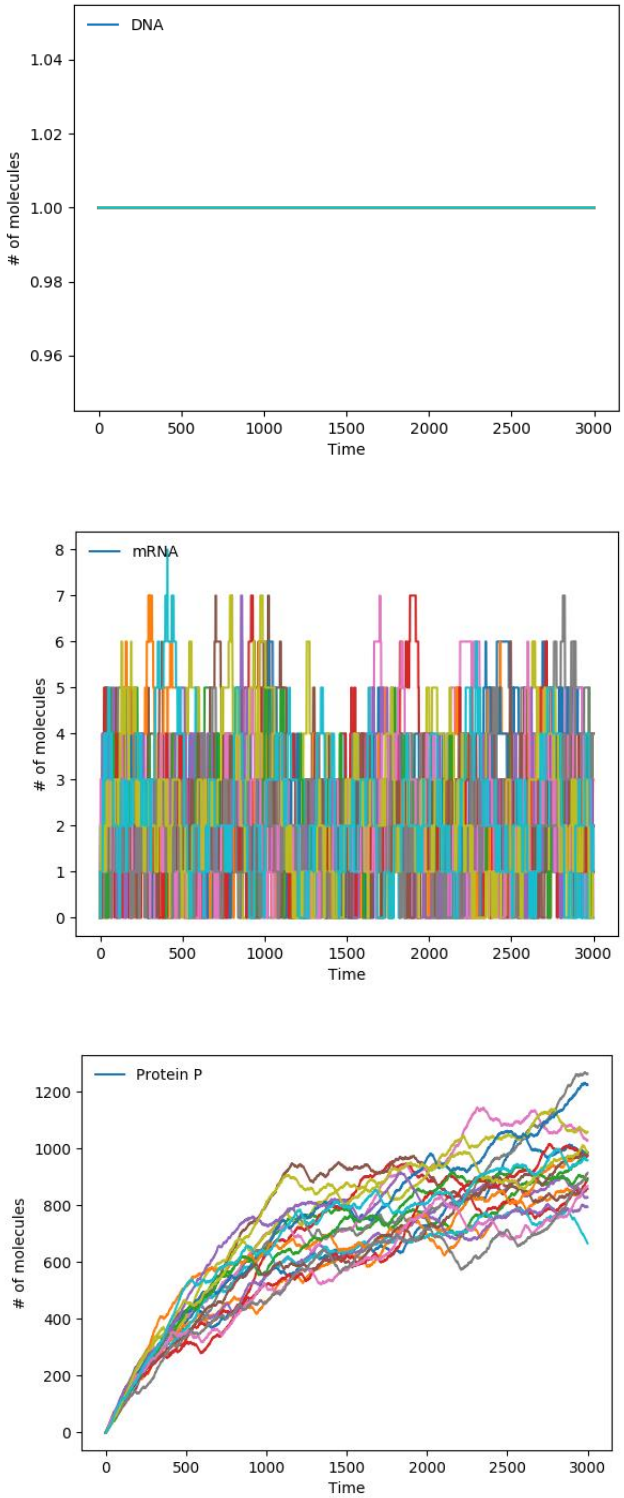

Figure 27: Molecular dynamics of DNA, mRNA and protein P for 20 cells and 3000 seconds

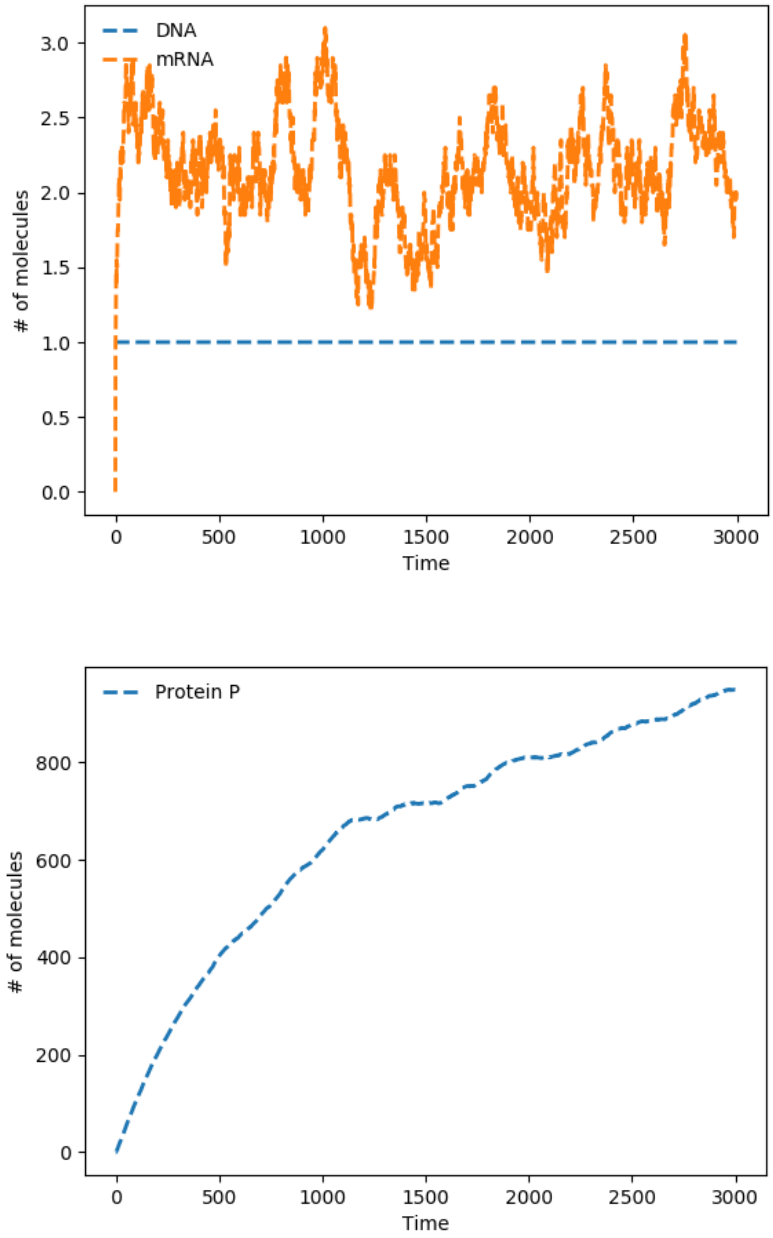

Figure 28: Average molecular dynamics of DNA, mRNA and protein P for 20 cells and 3000 seconds

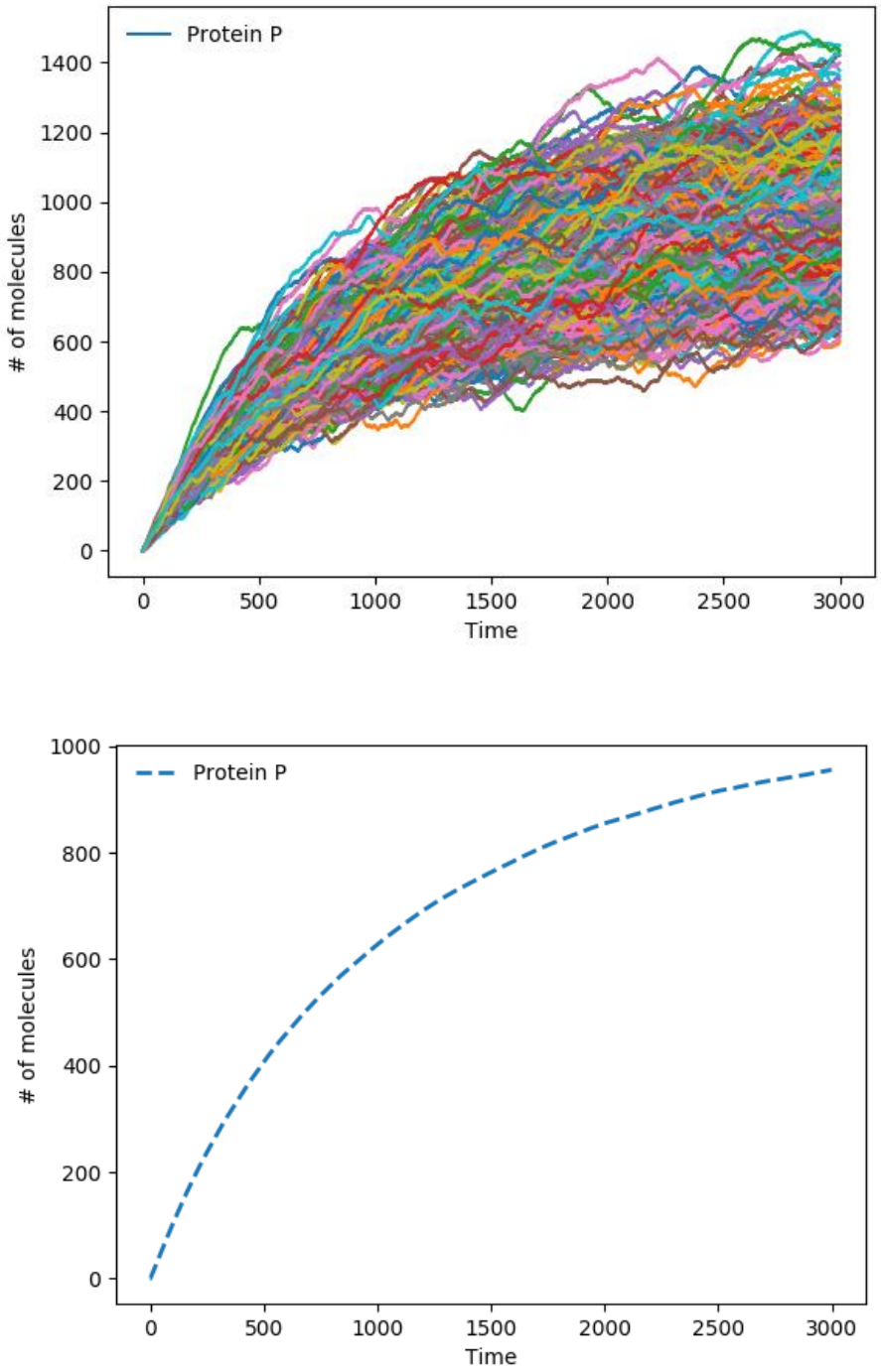

Figure 29: Molecular dynamics of P for 2000 cells and 3000 seconds. Dotted line represents the average of 2000 cells.

Example 2: Michaelis Menten Kinetics

```
begin model

# Parameter block
begin parameter

kf      1e-5
kr      0.1
kp      0.05

end parameter

# Species block
begin species

E
S
ES
P

end species

# Initial condition block
begin initial condition

E     5e3
S     1e4

end initial condition

# Reaction rules block
begin reaction rules

E + S -> ES         $ kf          $$ (constant)

ES -> E + S         $ kr          $$ (constant)

ES -> P + E         $ kp          $$ (constant)

end reaction rules


end model

simulate(N = 20, T = 400)
```

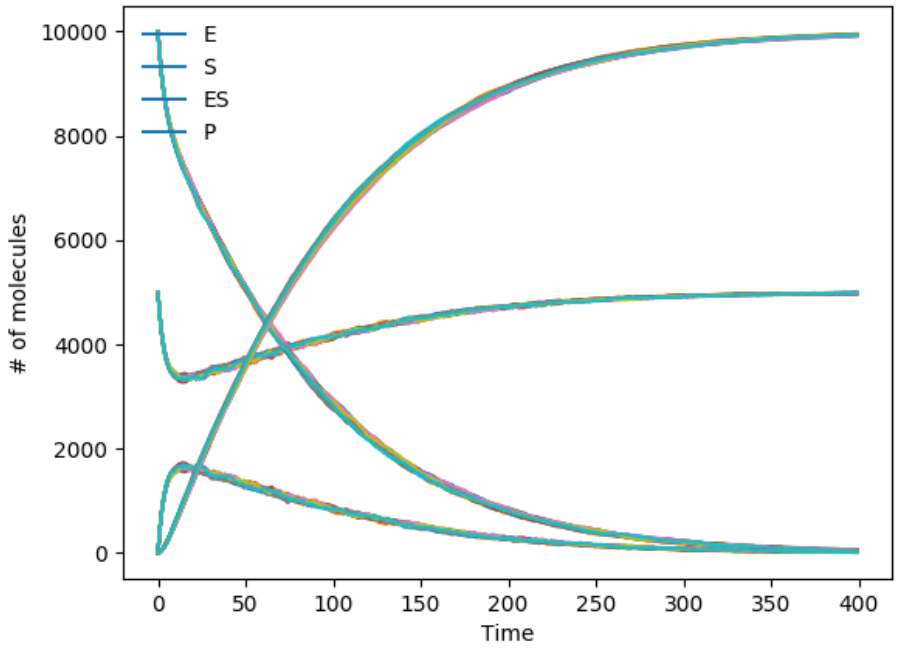

Figure 30: Molecular dynamics of Michaelis Menten kinetics for 20 cells and 400 seconds

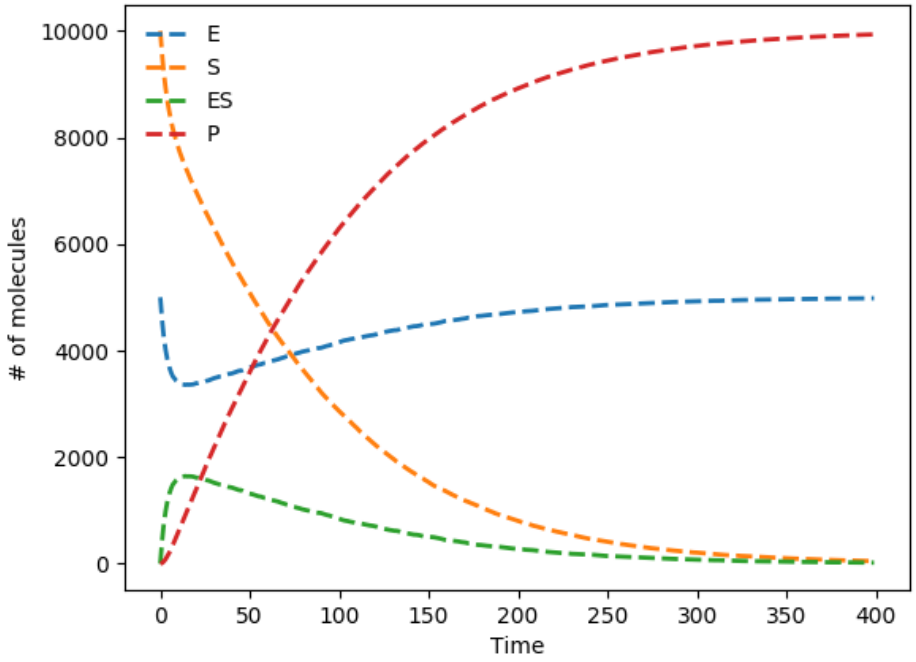

Figure 31: Average molecular dynamics of Michaelis Menten kinetics for 20 cells and 400 seconds

Example 3: Bacterial Quorum Sensing [2]

```
begin model

# Parameter block
begin parameter

dI          0.027       # Degradation of LuxI
f(x3,t)     218.98      # Protein expression of LuxI
dR          0.156       # Degradation rate of LuxR
ttLuxR      76.12       # Protein expression rate of LuxR
kf1         0.1         # Association constant of LuxR and AHL(IC)
kr1         10          # Unbinding rate of of LuxR.AHL(IC)
dc          0.017       # Degradation rate of (LuxR.AHL(IC))2
kf2         0.05        # Dimerization rate of LuxR.AHL(IC)
kr2         1           # Dissociation rate of dimer (LuxR.AHL(IC))2
dA          0.057       # Degradation rate of internal AHL(IC)
kA          0.04        # Synthesis rate of AHL(IC) by LuxI
D           2           # Diffusion rate of internal AHL(IC)
DVC         2.2e-6      # Diffusion rate of external AHL(EC)
dAe         0.04        # Degradation rate of external AHL(EC)

end parameter

# Species block
begin species

LuxI
LuxR
(LuxR.AHL(IC))2
AHL(IC)
LuxR.AHL(IC)
AHL(EC)

end species

# Initial condition block
begin initial condition

(LuxR.AHL(IC))2  10

end initial condition

# Reaction rules block
begin reaction rules

#1. degradation of LuxI

LuxI  ->              $ dI          $$ (constant)

#2. Production of LuxI
```

```
      ->   LuxI        $ f(x3,t)    $$ (constant)    $$$ DRR1
#3. Degradation of LuxR
LuxR    ->             $ dR         $$ (constant)
#4. Production of LuxI
        ->  LuxR       $ ttLuxR     $$ (constant)
#5. Association of LuxR and AHL(IC)
LuxR + AHL(IC)   ->   LuxR.AHL(IC)       $ kf1     $$ (constant)
#6. Dissociation of LuxR and AHL(IC)
LuxR.AHL(IC)      ->   LuxR + AHL(IC)    $ kr1     $$  (constant)
#7. Degradation of (LuxR.AHL(IC))2
(LuxR.AHL(IC))2  ->                      $ dc      $$ (constant)
#8. Dimerization of LuxR.AHL(IC)
LuxR.AHL(IC) + LuxR.AHL(IC) -> (LuxR.AHL(IC))2  $ kf2  $$ (constant)
#9. Dissociation of (LuxR.AHL)2
(LuxR.AHL(IC))2 -> LuxR.AHL(IC) + LuxR.AHL(IC)  $ kr2  $$ (constant)
#10. Degradation of autoinducer
AHL(IC)      ->                          $ dA       $$ (constant)
#11. Synthesis of AHL(IC)
LuxI     ->     LuxI  +  AHL(IC)         $ kA       $$ (constant)
#12. Diffusion of AHL(IC) from cell to environment
AHL(IC)      ->     AHL(EC)              $ D        $$ (constant)
#13. Diffusion of AHL(EC) from environment to cell
AHL(EC)  ->    AHL(IC)                   $ DVC      $$ (constant)
#14. Degradation of AHL(EC)
AHL(EC)   ->                             $ dAe      $$ (constant)
```

```
end reaction rules

# Dependent reaction block
begin dependent reaction rules

[DRR1] *= ((100 + 0.01 * moleculeA[(LuxR.AHL(IC))2]) / (100 +
moleculeA[(LuxR.AHL(IC))2]))

end dependent reaction rules


# Environmental species block
begin environmental species

AHL(EC)

end environmental species

end model


simulate(N = 20, T = 200, cores = 8, meanDivisionTime = 45)
```

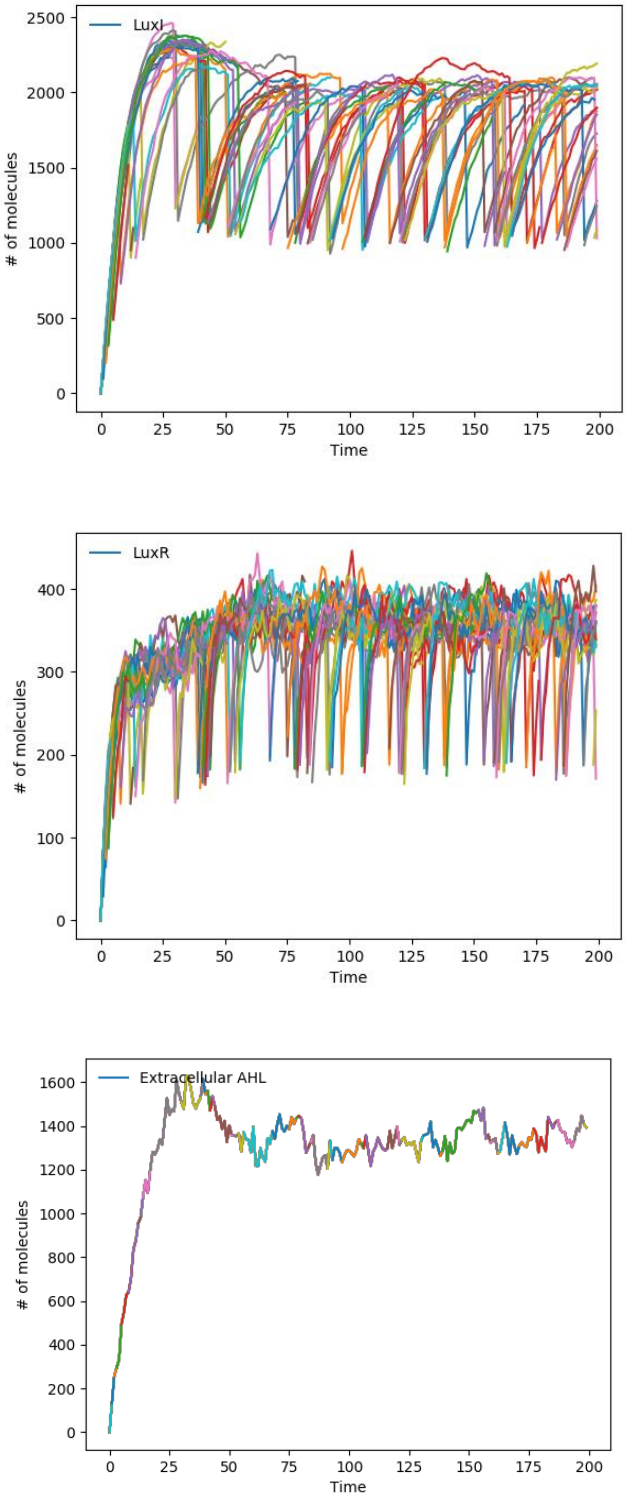

Figure 32: Molecular dynamics of LuxI, LuxR and external AHL for 20 cells and 200 min

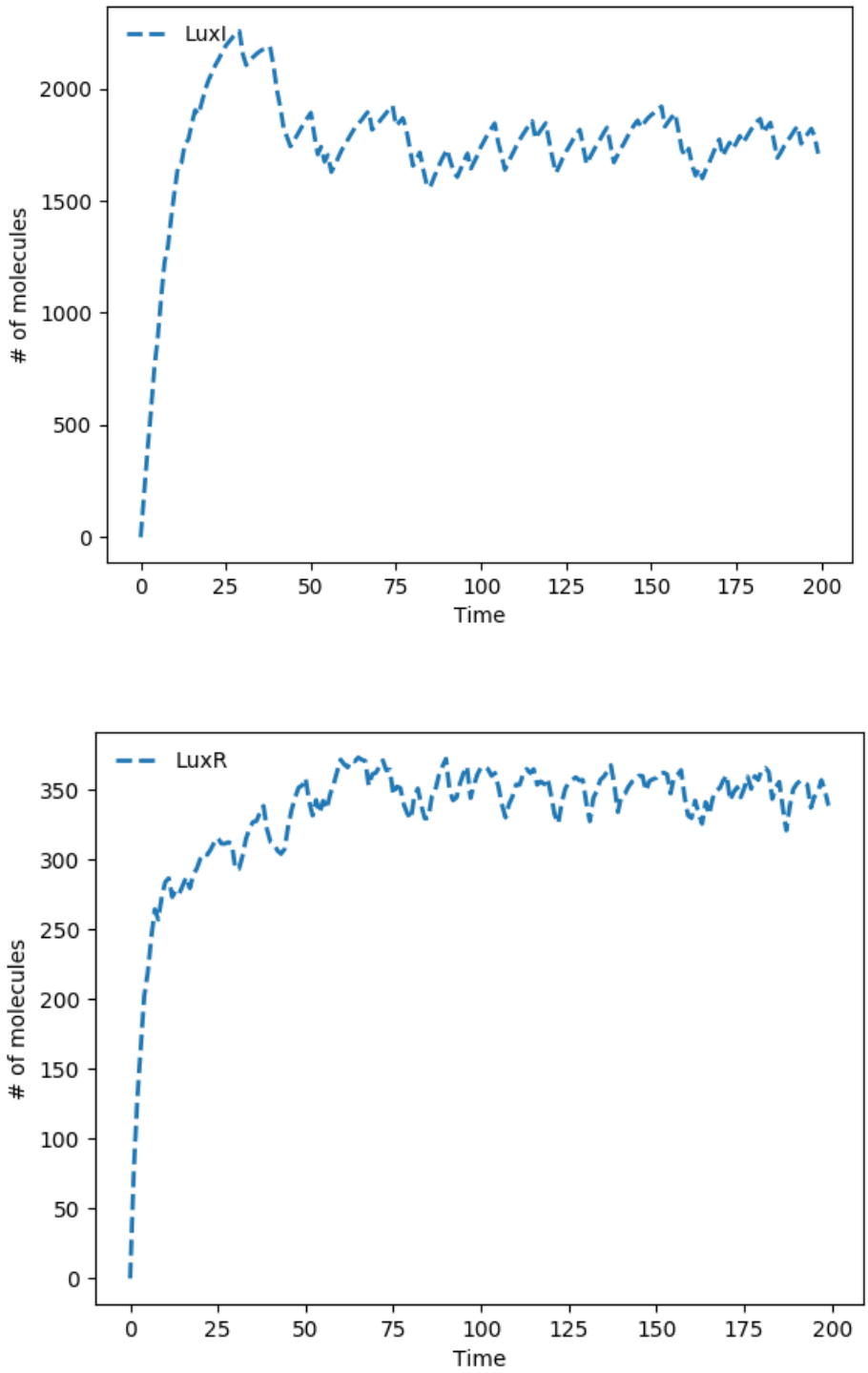

Figure 33: Average molecular dynamics of LuxI and LuxR for 20 cells and 200 min

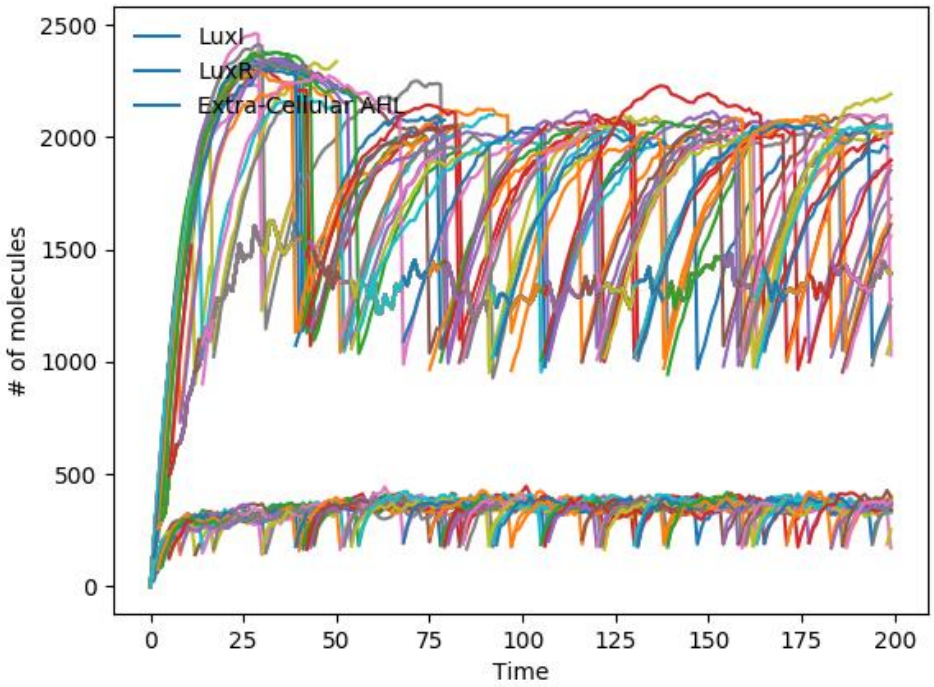

Figure 34: Molecular dynamics of LuxI, LuxR and external AHL for 20 cells and 200 min

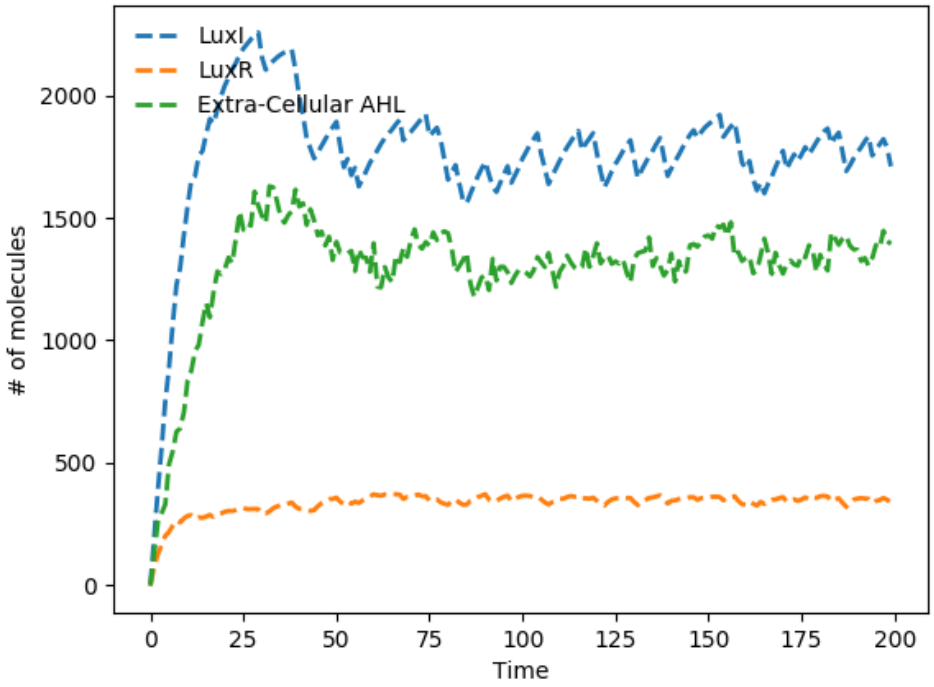

Figure 35: Average molecular dynamics of LuxI, LuxR and external AHL for 20 cells and 200 min

Note: Example models are available in 'GUI/BIBM/model' folder.

References

1) Satyaki Roy, Mohammad Aminul Islam, Dipak Barua and Sajal Das. "A Scalable Parallel Framework for Multicellular Communication in Bacterial Quorum Sensing". In Proceedings of the 11th EAI International Conference on Bio-inspired Information and Communications Technologies (accepted), 2019.
2) Islam, Mohammad, et al. "Multicellular Models Bridging Intracellular Signaling and Gene Transcription to Population Dynamics." Processes 6.11 (2018): 217.
3) Weber, Marc, and Javier Buceta. "Dynamics of the quorum sensing switch: stochastic and non-stationary effects." BMC systems biology 7.1 (2013): 6.